MULTI-LAYER DESIGNS AND COMPOSITE GAUSSIAN PROCESS MODELS WITH ENGINEERING APPLICATIONS

A Thesis
Presented to
The Academic Faculty

by

Shan Ba

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology
August 2012

MULTI-LAYER DESIGNS AND COMPOSITE GAUSSIAN PROCESS MODELS WITH ENGINEERING APPLICATIONS

Approved by:

Roshan Joseph Vengazhiyil, Advisor
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Jianjun Shi
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

C. F. Jeff Wu
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

Santanu Dey
H. Milton Stewart School of Industrial and Systems Engineering
Georgia Institute of Technology

William Brenneman
Statistics Department
The Procter & Gamble Company

Date Approved: 4 May 2012

To my beloved parents and my fiancée

for their support, inspiration and encouragement

during this challenging journey.

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my appreciation to all who have influenced, stimulated, expedited, and warmly supported my work in various ways during my doctoral studies at Georgia Tech.

First and foremost, it is my immense pleasure to express my deep and sincere gratitude to my advisor, Professor Roshan Joseph Vengazhiyil, for his amazing guidance and incredible assistance in all phases of my doctoral program. I can still clearly remember the moment when he recruited me as his student in 2007, which has dramatically changed my life and brought me to the amazing world of academia! Since then, he has offered me overwhelming support in every possible way I can think about. He is not only my academic advisor, but also a great mentor for my life.

I am also extremely grateful to Professor C. F. Jeff Wu for his guidance on my research and active support during my studies. His knowledge, insight and inspiration have greatly influenced me.

I would like to thank Dr. William Brenneman and Dr. William Myers for their valuable mentorship when I served as the research assistant for the Procter and Gamble Company. I have been constantly inspired and amazed by their unrivalled experience in statistical practice. I'm also thankful to Professor Jianjun Shi and Professor Santanu Dey for their help on my research and for serving on my dissertation committee.

I thank all my classmates and friends who shared time and knowledge with me at Georgia Tech. They made my graduate life wonderful and enjoyable. I consider myself very fortunate to be able to work together with these outstanding people.

Last, but by no means the least, my heartfelt appreciation and gratitude goes

to my beloved family, especially my parents and my fiancée Taoran Dong, for their constant support and encouragements. This thesis would not be possible without them!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

The modern era witnesses the prosperity of computer experiments, which play a critical role in many fields of technological development where the traditional physical experiments are infeasible or unaffordable to conduct. By developing sophisticated computer simulators, people are able to evaluate, optimize and test complex engineering systems even before building expensive prototypes. Since the computer experiments are usually time-consuming to run, surrogate models are often fitted to approximate these computationally expensive simulations. Because the fitted surrogate models are much faster to run, they can be readily used to provide instant predictions and facilitate the analysis of the underlying system.

In building the surrogate model for computer experiments, there are two important research topics. The first one is how to efficiently select a set of input values to run the computer simulation for a finite number of times, and this is called the *design* of computer experiments. After we obtain the simulation outputs, the second question is how to *model* these data in order to accurately approximate the unknown response surface generated by the simulator.

This thesis consists of three chapters, covering topics in both the design and modeling aspects of computer experiments as well as their engineering applications. The first chapter systematically develops a new class of space-filling designs for computer experiments, and the second chapter proposes a novel modeling approach for approximating computationally expensive functions that are not second-order stationary. The third chapter is devoted to a two-stage sequential strategy which integrates analytical models with finite element simulations for a micromachining process.

In computer experiments, space-filling designs such as Latin hypercube designs (LHDs) are widely used. However, finding an optimal LHD with good space-filling properties is computationally cumbersome. On the other hand, the well-established factorial designs in physical experiments are unsuitable for computer experiments owing to the redundancy of design points when projected onto a subset of factor space. In the first chapter, we present a new class of space-filling designs developed by splitting two-level factorial designs into multiple layers. The method takes advantages of many available results in factorial design theory and therefore, the proposed Multi-layer designs (MLDs) are easy to generate. Moreover, our numerical study shows that MLDs can have better space-filling properties than optimal LHDs.

In the second chapter, a new type of non-stationary Gaussian process model is developed for approximating computationally expensive functions. The new model is a composite of two Gaussian processes, where the first one captures the smooth global trend and the second one models local details. The new predictor also incorporates a flexible variance model, which makes it more capable of approximating surfaces with varying volatility. Compared to the commonly used stationary Gaussian process model, the new predictor is numerically more stable and can more accurately approximate complex surfaces when the experimental design is sparse. In addition, the new model can also improve the prediction intervals by quantifying the change of local variability associated with the response. Advantages of the new predictor are demonstrated using several examples.

Chapter three considers the problem of integrating analytical models with finite element simulations. We show that computationally cheap analytical models can be used to perform a sensitivity analysis which can reveal critical information about the underlying system prior to conducting the computationally intensive simulation study. We propose a two-stage sequential strategy, which can efficiently absorb the prior information from the sensitivity analysis and assign a customized number of

levels for each input variable in the finite element simulations. The method is also broadly applicable for integrating other types of models having different levels of accuracy and speed. A case study for developing force metamodels in micromachining is presented to illustrate the effectiveness of the proposed method.

CHAPTER I

MULTI-LAYER DESIGNS FOR COMPUTER EXPERIMENTS

1.1 Introduction

Computer experiments play a major role in the modern era of scientific and technological development. For example, airbags in a car can be designed through sophisticated computer simulation that mimics a car-crash in a computer instead of building and crashing real cars. This results in substantial savings of time and cost for the automobile manufacturer. There are many other successful applications of computer experiments. See the books by Santner, Williams and Notz (2003) and Fang, Li and Sudjianto (2006). However, computer experiments can also be time-consuming and expensive, although not as expensive as the real physical experiments. Thus, it is important to carefully design a computer experiment and analyze the data so that maximum information about the system can be gathered.

Latin hypercube designs (LHDs) (McKay, Beckman and Conover 1979) are commonly employed in designing computer experiments. Because a randomly generated LHD can be poor in terms of space-filling, most of the research in this area has focused on finding optimal LHDs. For examples, orthogonal array-based LHD (Owen 1992; Tang 1993), maximin LHD (Morris and Mitchell 1995), orthogonal LHD (Ye 1998) and orthogonal-maximin LHD (Joseph and Hung 2008) are just a few of them. Optimal designs are usually found using some optimization algorithms such as simulated annealing (Morris and Mitchell 1995) and other stochastic evolutionary algorithms (Jin, Chen and Sudjianto 2005). However, these algorithms are slow and may not find the global optimum, particularly when the number of runs and/or the number of factors are large. Therefore, when the algorithms are terminated after a reasonable amount of time, we may end up in a local optimum.

On the other hand, a significant amount of knowledge about optimal factorial designs is already available in the physical experiments' literature (Hedayat, Sloane and Stufken 1999; Box, Hunter and Hunter 2005; Mukerjee and Wu 2006; Wu and Hamada 2009). However, these designs are not popular in computer experiments mainly due to the redundancy of the design points when projected onto a subspace, i.e., some runs get replicated when some of the factors turn out to be insignificant during the data analysis. Such replications are not useful in computer experiments because of the deterministic nature of the outputs (no random error). In this work, we attempt to convert the optimal factorial designs into space-filling designs and make them suitable for computer experiments. By taking advantage of the geometric properties of factorial designs, we can significantly reduce the computational time for finding optimal space-filling designs. Similar ideas related to geometrical constructions have been employed by Steinberg and Lin (2006) for efficiently constructing orthogonal LHDs. See also the work by Bingham, Sitter and Tang (2009) and Lin, Bingham, Sitter and Tang (2010). However, good space-filling of design points is considered more important than orthogonality of the factorial effects in the investigation of highly complex functions. This is where the proposed designs exhibit some advantages.

This chapter is organized as follows. In Section 1.2, we present a new type of space-filling design, developed by splitting the two-level factorial design into multiple layers. In Section 1.3, we discuss how to choose the initial two-level factorial design. In Section 1.4, we propose a general strategy to split the design optimally into several layers. Section 1.5 is devoted to the discussion of how many layers should be used, how many points should be allocated to each layer, and how should the layers be spaced. In Section 1.6, we conduct numerical studies and show that the proposed designs can have better space-filling properties than optimal LHDs. Section 1.7 extends the design into more flexible run sizes and some final concluding remarks are given in Section 1.8.

1.2 Concepts of Multi-Layer Design

Suppose we would like to construct an experimental design in n runs for p factors. Let $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$ be the experimental design, where each $\boldsymbol{x}_i \in \mathcal{X} = [-1, 1]^p$. Two of the most popular space-filling criteria for designing a computer experiment are the *maximin* and *minimax* (distance) criteria (Johnson, Moore and Ylvisaker 1990). The maximin criterion tries to spread out the points in $\mathcal{X}$ so that the minimum distance among the design points is maximized. Thus, the maximin design (D_{Mm}) can be defined as

$$\min_{\boldsymbol{x}_i, \boldsymbol{x}_j \in D_{Mm}} d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \max_{D} \min_{\boldsymbol{x}_i, \boldsymbol{x}_j \in D} d(\boldsymbol{x}_i, \boldsymbol{x}_j), \tag{1}$$

where $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the distance between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. In contrast, the minimax criterion tries to spread out the points in $\mathcal{X}$ so that the maximum distance from any point $\boldsymbol{x} \in \mathcal{X}$ to the design is minimized. Thus, the minimax design ensures that no point in $\mathcal{X}$ is far away from a design point. Suppose we define the distance between an arbitrary point $\boldsymbol{x} \in \mathcal{X}$ to the design D by $d(\boldsymbol{x}, D) = \min_{\boldsymbol{x}_i \in D} d(\boldsymbol{x}, \boldsymbol{x}_i)$. Then the minimax design (D_{mM}) can be formally defined as

$$\max_{\boldsymbol{x} \in \mathcal{X}} d(\boldsymbol{x}, D_{mM}) = \min_{D} \max_{\boldsymbol{x} \in \mathcal{X}} d(\boldsymbol{x}, D). \tag{2}$$

For $n = 2^p$ experimental runs (where p is an integer), we can use geometry to find the maximin and minimax designs. An example of four-run design in two factors ($n = 4, p = 2$) is shown in Figure 1. However, in general, finding maximin and minimax designs for an arbitrary n is very complicated. The case of minimax designs is most challenging, since an optimization step is required to calculate the maximum distance.

Interestingly, both the designs (a) and (b) in Figure 1 are full factorial 2^2 designs, but with different scales. Which design should be used in the experiment? To answer this question, we also need to look at the modeling of data. It is quite common to model the computer experiment data by using *kriging* (Sacks, Welch, Mitchell and Wynn 1989). A *universal kriging* model is the sum of a linear model part and a

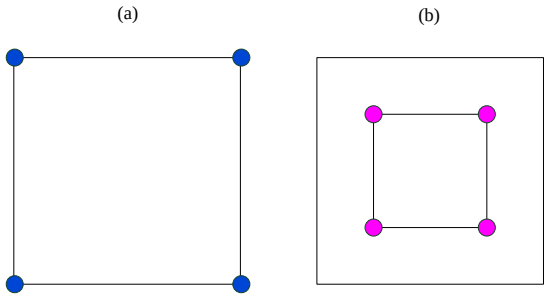

Figure 1: Four-run design in two factors: (a) Maximin design. (b) Minimax design.

kriging model part and is given by

$$y(\boldsymbol{x}) = \mu(\boldsymbol{x}) + Z(\boldsymbol{x}), \qquad \boldsymbol{x} \in \mathbb{R}^p, \tag{3}$$

where $\mu(\boldsymbol{x}) = \sum_{i=0}^{m} \beta_i f_i(\boldsymbol{x})$, $f_i(\boldsymbol{x})$'s are some known functions, β_i's are some unknown parameters, and $Z(\boldsymbol{x})$ is a stationary stochastic process with mean 0 and covariance function $cov\{Z(\boldsymbol{x}+\boldsymbol{h}), Z(\boldsymbol{x})\} = \sigma^2 R(\boldsymbol{h};\boldsymbol{\theta})$. Here $\boldsymbol{\theta} \in \mathbb{R}^p$ is a vector of the unknown correlation parameters. It is also possible to relax the assumption that $f_i(\boldsymbol{x})$'s are known by selecting them from a candidate set of functions using a variable selection technique (Joseph, Hung and Sudjianto 2008). They call it a *blind kriging* model, which is shown to give improved prediction over universal kriging and *ordinary kriging*. (In ordinary kriging only a constant is used in the mean part $\mu(\boldsymbol{x})$.)

The linear model part $\mu(\boldsymbol{x}) = \sum_{i=0}^{m} \beta_i f_i(\boldsymbol{x})$ in (3) captures the global trend, whereas the kriging model part $Z(\boldsymbol{x})$ in (3) captures the local trend. As shown by the success of blind kriging in Joseph et al. (2008), both parts are important for prediction. Thus, good experimental designs should be effective for estimating both the linear model part and the kriging model part. In the 2^2 design example above, if we use a linear model with the two main effects and the two-factor interaction, then clearly the maximin design in Figure 1(a) is better, because the design can be shown to be universally optimum (Kiefer 1975). On the other hand, if we would like to estimate the kriging model accurately, then the design in Figure 1(b) can be shown to work better in the sense that it minimizes the maximum prediction variance (which has a theoretical justification via the asymptotic results on G-optimality in Johnson et al. 1990). Thus, for estimating both the global and local trends efficiently, we

should consider a compromise between the two types of designs. This is the major motivation in proposing a new class of designs for computer experiments.

Consider an alternative design in Figure 2. Here two design points on the boundary belong to the maximin design, but the other two points have been pulled to the interior to coincide with part of the minimax design. This can be considered as a compromise design. Another way to interpret the new design is to visualize it as having two layers with the design points placed on the two layers. The layers and the placement of the points are judiciously chosen to obtain optimal space-filling properties for the design. The concept can also be generalized into any number of dimensions and any number of runs. In fact, we can have more than two layers and can spread out the points more evenly. We call this new class of designs as *multi-layer designs* (MLD). Piepel, Anderson and Redgate (1993) has used a similar two-layer structure to construct response surface designs for irregularly-shaped regions. However, their layered designs are model-dependent and are only suitable for fitting low-order polynomial models on irregular experimental regions. Thus, their work is fundamentally different from the proposed MLD which is used as a new type of space-filling design in computer experiments.

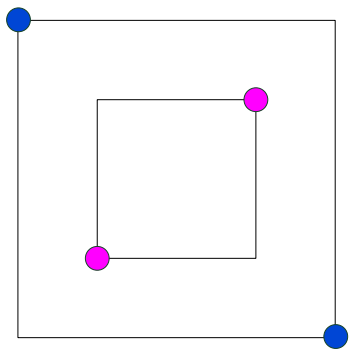

Figure 2: MLD: a compromise between maximin and minimax designs.

As another example of MLD, consider the eight-run design in three factors. Figure 3 shows the maximin and minimax designs, both of which are also full factorial designs. Figure 4(a) shows a two-layer design and Figure 4(b) shows a three-layer design. If we continue to divide the layers, finally we can also obtain a four-layer

design as shown in Figure 4(c). We compared this four-layer design with 1000 randomly generated LHDs. Figure 5 shows the density plots of the minimum distance (among the design points) and the maximum distance (from the design to a point in the experimental region). We can see that the MLD in this example performs even better than the best LHD for both measures.

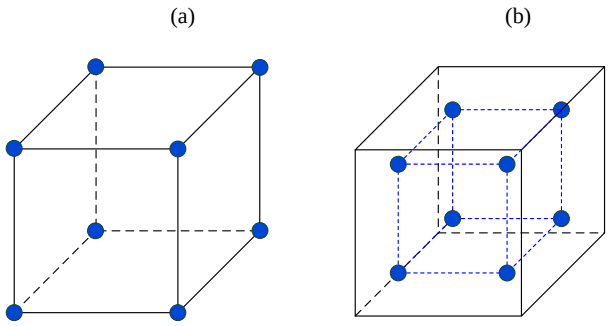

Figure 3: Eight-run design in three factors: (a) Maximin design. (b) Minimax design.

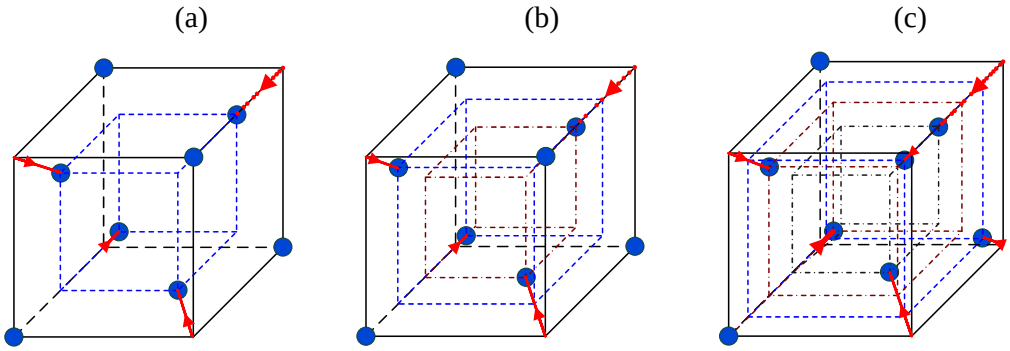

Figure 4: MLDs: (a) Two-layer design (b) Three-layer design (c) Four-layer design.

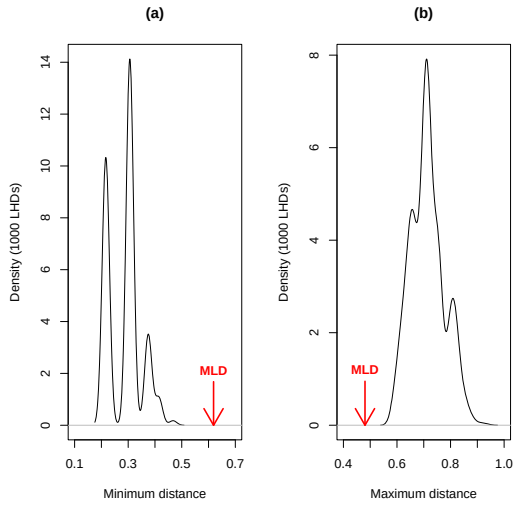

Figure 5: Density plots for 1000 LHDs: (a) Minimum Distances (b) Maximum Distances.

1.3 Choosing Base Designs

Previously we have seen that MLDs can be constructed from the two-level full or fractional factorial designs, and we call them *base designs* for MLDs. However, for given number of factors and run size requirement, there are many possible choices for the base design. In this section, we discuss which criteria should be used to discriminate these designs and which specific design should be chosen as the base design.

Generally, suppose we have p factors each at two levels, a two-level full factorial design consists all the possible $2 \times 2 \times \cdots \times 2 = 2^p$ runs, and is often referred to as a 2^p design. For run size economy, the two-level fractional factorial design is commonly used in physical experiments, which can be referred to as a 2^{p-k} design, since it is a 2^{-k}th fraction of the 2^p full factorial design and consists 2^{p-k} runs (where $k > 0$ and k is an integer). The fraction is determined by k *defining words*, where a word consists of letters which are the names of the factors denoted by $1, 2, \ldots, p$. The number of letters in a word is its *wordlength* and the group formed by the k defining words is called the *defining contrast subgroup*, which consists of $2^k - 1$ words plus the identity element $\mathbf{I}$. If we let A_i denote the number of words of length i in the defining contrast subgroup,

the vector $W = (A_1, A_2, \ldots, A_p)$ is called the *wordlength pattern* of the 2^{p-k} design. To discriminate between the many possible 2^{p-k} designs and identify good ones, Fries and Hunter (1980) proposed the following *minimum aberration* criterion. For any two 2^{p-k} designs d_1 and d_2, let s be the smallest integer such that $A_s(d_1) \neq A_s(d_2)$. Then d_1 is said to have less aberration than d_2 if $A_s(d_1) < A_s(d_2)$. If there is no design with less aberration than d_1, then d_1 has minimum aberration. In literatures such as Wu and Hamada (2009), the generators of minimum aberration 2^{p-k} designs have been systematically tabulated.

Although minimum aberration criterion is commonly used in physical experiments, under the settings of computer experiments the space-filling property of a design is considered to be more important, and good 2^{p-k} designs should be selected in this sense. Fortunately, it has already been found that within the class of two-level factorial designs, many minimum aberration 2^{p-k} designs and maximin 2^{p-k} designs coincide (Kerr 2001; Kwong 2004). Even when they mismatch, the minimum aberration designs still tend to perform well with respect to their maximin distance rankings. In addition, Fang and Mukerjee (2000) established a strong theoretical connection between the aberration and another space-filling criterion: the uniformity measure. They proved that, for any 2^{p-k} design, the centered L_2-discrepancy measure of uniformity in $[0,1]^p$ is $\{CL_2\}^2 = (\frac{13}{12})^p - 2(\frac{35}{32})^p + (\frac{9}{8})^p\{1 + \sum_{r=1}^{p} \frac{A_r}{9^r}\}$, where $A_r, r = 1, \ldots, p$ are the wordlength pattern of the design. Since the coefficient of A_r decreases exponentially with r, minimum aberration 2^{p-k} designs are almost the most uniform 2^{p-k} designs. Therefore, among all two-level factorial designs that place points on the 2^p corners of design region, minimum aberration 2^{p-k} designs possess the most favorable space-filling properties, and they can be readily used as good base designs for constructing MLDs. This approach enables us to utilize the vast amount of literature on design theory for physical experiments (Mukerjee and Wu 2006; Wu and Hamada 2009). In the following sections, we will present a general strategy to construct MLDs based on the chosen two-level factorial designs.

1.4 Splitting Design Points Optimally Into Layers

An essential step in constructing MLDs is to split the base design points optimally into several subgroups in order to allocate each of them to different layers. The main idea in splitting design points is related to the well-known foldover technique (Box and Hunter 1961; Li and Lin 2003; Box et al. 2005; Wu and Hamada 2009). This commonly used strategy considers adding follow-up experiments by reversing signs of one or more columns in the initial design. However, instead of doubling the experimental runs, we propose a backward procedure to halve the number of runs each time. Specifically, given a 2^{p-k} design, we split it into two parts in such a way that each of them forms the foldover plan of the other. The resulting two 2^{p-k-1} designs are called *half-designs* and the previous 2^{p-k} design called *original design*. Similarly, each of the 2^{p-k-1} designs can again be further divided into two half-designs; thereby totally producing four small designs each with 2^{p-k-2} points. This procedure can be continued, and finally we split the original design into several small two-level fractional factorial designs which possess the following property: if we combine them together by folding over each pair iteratively, the original 2^{p-k} design can be recovered. In the following subsections, we first develop a general construction method for half-designs, and then discuss how to obtain the optimal half-designs.

1.4.1 Construction of Half-Designs

For a 2^{p-k} design, we refer to its first $p-k$ factors as *basic factors*, whose columns constitute a 2^{p-k} runs full factorial design, and the last k factors as *generated factors*, whose columns are determined by the k defining words and can be generated from the first $p-k$ independent columns. A defining word for a generated factor is also called a *generator*, and these two terms are used interchangeably in this chapter. The general construction procedure for half-designs comprises the following steps:

Step 1. Obtain a 2^{p-k} design as the original design.

Step 2. Define a new generator for a chosen basic factor, and keep all previous generators unchanged. If the previous generators also contain the chosen factor,

update them by substituting the chosen factor with its new defining word. Call the resulting 2^{p-k-1} design the first half-design.

Step 3. Same as in step 2, but choose the opposite sign for the new generator. Obtain a second half-design.

In the above procedure, since each time a basic factor becomes a generated factor, the design run size is halved. This method can also be applied iteratively to split a design into many parts. Note that these steps establish a rather general framework, since in step 1 the original design can be an arbitrary 2^{p-k} design. In addition, any basic factor can be chosen in step 2 and there are also many ways to define the new generator. Discussions on their optimal choices are deferred to the next subsection. Here we first provide proofs to show that the two half-designs are foldover plans of each other, and the original design can always be recovered. The following Lemma 1 reveals an important structure in their defining contrast subgroups.

Lemma 1. *The defining contrast subgroup of half-designs produced in the general construction procedure consists three parts:*

Part (i) *All words in the defining contrast subgroup of the original design.*

Part (ii) *Each word in part* (i) *times the new defined generator.*

Part (iii) *The new defined generator.*

In sum, there are $(2^k - 1) + (2^k - 1) + (1) = 2^{k+1} - 1$ *words in the defining contrast subgroup of each half-design.*

The proof for Lemma 1 is straightforward and omitted here. We need to point out that this special structure does provide us a very useful shortcut in generating the defining contrast subgroup of half-designs. According to Lemma 1, the half-design can inherit all the $2^k - 1$ words from original design right away, and we only need to perform additional $2^k - 1$ calculations. This almost reduces the computation by half compared to calculating all the $2^{k+1} - 1$ words directly. Using this lemma, we can also prove the following result.

Theorem 1. *In the general construction procedure, the second half-design is the foldover plan of the first half-design, and their combined design is the original design in step 1.*

Proof. From the general procedure, it can be seen that the two half-designs are constructed from the same original design, by adding a new generator with opposite signs. As a result, all defining words in their defining contrast subgroups are the same, except for their signs. Therefore, the second half-design can be obtained from the first half-design by reversing signs of some columns, and they are foldover plans of each other. In addition, according to Lemma 1, the first and second half-designs have exactly the same words in part (i). However, since they choose opposite signs for the new defined generator, signs of all their words in parts (ii) and (iii) are opposite. Therefore, when the two half-designs are combined together, the defining words in parts (ii) and (iii) cancel out and only the words in part (i) remain. This shows that the combined design have exactly the same defining contrast subgroup with the original design in step 1. Therefore, they are identical. ♢

Now we use an example to illustrate the construction of half-designs and how their defining contrast subgroups can be quickly generated. Suppose we choose original design as the minimum aberration 2^{8-3} design, whose generators $6 = 123$, $7 = 124$, $8 = 2345$ can be obtained from literatures such as Wu and Hamada (2009). Its defining contrast subgroup has seven words: $I = 1236 = 1247 = 23458 = 3467 = 14568 = 13578 = 25678$. Next suppose we add a new generator $5 = 134$. Since factor 5 also appears in the generator $8 = 2345$, an update is needed by substitution: $8 = 234(134) = 12$. Through this way, we can obtain the first half-design with generators $5 = 134$, $6 = 123$, $7 = 124$, $8 = 12$. Note that its fifteen defining words do not need to be calculated from these four generators directly. By the shortcut in Lemma 1, the defining contrast subgroup can be obtained in three parts, in which the first part just inherits the whole defining contrast subgroup of the original design:

I=1236=1247=23458=3467=14568=13578=25678

[Part (i), words from the original 2^{8-3} design]

=2456=2357=128=1567=368=478=1234678

[Part (ii), each word in part (i) times 1345]

=1345 [Part (iii), the new defining word added for factor 5]

Similarly, the new generator added for the second half-design is $5 = -134$, which takes the opposite sign to that in the first half-design. As a result, the four generators of the second half-design are $5 = -134$, $6 = 123$, $7 = 124$, $8 = -12$. Its defining contrast subgroup can also be obtained in three parts through the shortcut in Lemma 1:

I=1236=1247=23458=3467=14568=13578=25678 [Part (i)]

=–2456=–2357=–128=–1567=–368=–478=–1234678 [Part (ii)]

=–1345 [Part (iii)]

By now, it is obvious that when we combine these two half-designs together, all their words in part (ii) and part (iii) will be canceled out and only the words in part (i) remain. Therefore, their combined design is just the original 2^{8-3} design. If we apply this construction procedure repeatedly, the 2^{8-3} design can be split into many more parts.

1.4.2 Optimal Half-Designs

In the general procedure to construct half-designs, we need to choose a basic factor in step 2 and also define a new generator for it. However, there are $p-k$ basic factors to choose; and for each chosen basic factor, there are also 2^{p-1} possible ways to define its new generator, since the new generator can contain all the other $p-1$ factors. Different choices can lead to many possible half-designs, and among them we define optimal half-designs as those that are optimal with respect to a chosen criterion. In this chapter, we use minimum aberration as the optimal criterion for half-designs, since these designs possess favorable space-filling properties as discussed at the end of Section 1.3.

The following lemma shows that when minimum aberration criterion is used, the optimal half-design of a 2^p full factorial design is apparent. The proof is straightforward and omitted.

Lemma 2. *For a 2^p full factorial design, its minimum aberration half-designs are the 2^{p-1} fractional factorial designs generated by $I = \pm 123 \cdots p$.*

In general, finding optimal half-designs of an arbitrary 2^{p-k} design by definition involves checking all the $(p-k) \times 2^{p-1}$ possible half-designs. Note, however, that many of these half-designs are equivalent. The following theorem shows that all the $(p-k) \times 2^{p-1}$ possible half-designs are equivalent to a much smaller subset of them, containing only $2^{p-k} - 1$ half-designs. By this result we can significantly reduce the computational effort in finding optimal half-designs.

Theorem 2. *Any possible half-design of a 2^{p-k} design is equivalent to one of the $2^{p-k} - 1$ half-designs constructed as follows: choose the ith basic factor in the general construction procedure, and each time define it as generated by each possible subset of the first $i-1$ factors, for $i = p-k, \ldots, 2, 1$.*

Proof. In the general construction procedure, for any chosen basic factor, its new generator can have 2^{p-1} possible choices, since this generator can include all the other $p-1$ factors. If the new defined generator contains generated factors, we can substitute these generated factors with their own defining words, and obtain a standardized form of the new generator which only includes basic factors. Therefore, any of these 2^{p-1} possible generators is equivalent to one of the 2^{p-k-1} standardized generators.

In addition, for $i = p-k, \ldots, 2, 1$, if the generator added to the ith basic factor also contains the hth basic factor ($h > i$), we can permutate the factor orders within this generator, and represent it equivalently as another generator added to the hth basic factor which also contains the ith factor. This shows that, when defining a new generator, we only need to consider including a subset of basic factors before the chosen factor.

As a result, when the ith factor is chosen, the new generator has 2^{i-1} possible choices. In sum, the total amount of possible half-designs reduces to $2^{p-k-1} + 2^{p-k-2} + 2^{p-k-3} + \cdots + 2 + 1 = 2^{p-k} - 1$. ♢

According to Theorem 2, the optimal half-design can be found by performing the general construction procedure for only $2^{p-k}-1$ times, where $2^{p-k}-1$ is usually a small value. Moreover, since minimum aberration is chosen as the optimality criterion, Lemma 1 provides another shortcut in comparing word length pattern of those $2^{p-k}-1$ half-designs. According to Lemma 1, all their defining words in part (i) are identical, and we only need to count their word lengths in part (ii) and (iii), which further improves the computation efficiency. In fact, for moderate-size original designs, we are able to obtain the optimal half-designs easily by hand; even for large problems, optimal half-designs can still be found instantly with a short computer code. We can also split the design points optimally into multiple layers, by iterating this method for a few more times.

Consider again the minimum aberration 2^{8-3} design generated by 6 = 123, 7 = 124, 8 = 2345. We now illustrate how to find its minimum aberration half-designs. In previous subsection, we show that adding a new generator 5 = 134 can lead to the half-design with generators 5 = 134, 6 = 123, 7 = 124, 8 = 12, and its corresponding word length is (0, 0, 3, 7, 4, 0, 1, 0). To find optimal half-design, we need to check each of the $2^{8-3}-1 = 31$ possible half-designs in a similar manner, each time by adding one of the following new generators: (1) 5=I, 1, 2, 3, 4, 12, 13, 14, 23, 24, 34, 123, 124, 134, 234, 1234, (2) 4=I, 1, 2, 3, 12, 13, 23, 123, (3) 3=I, 1, 2, 12, (4) 2=I, 1, (5) 1=I. Using the shortcut in Lemma 1, we can easily compare their word length patterns by hand, and the half-design presented in last subsection is found to be the one with minimum aberration. The second-best optimal half-design has word length pattern (0, 0, 4, 6, 4, 0, 0, 1), and can be obtained by adding new generator 4 = 13.

1.5 MLD Layouts

Given the single-layer base design, MLD can be constructed by iterating the procedure from previous section: each time splitting the design points into two optimal half-designs and moving half points inward as a new layer. To complete this construction process, we still need to answer two more questions: (i) how many layers should be used and how many points should be allocated to each layer? (ii) how should the

layers be spaced?

Regarding the first question, we propose a strategy to uniformly split the n-run 2^{p-k} base design points into $n/2$ layers, with exactly two points on each layer. To implement this strategy, in each step every current layer is split optimally into another two layers, and this process is repeated so that in the end all layers contain only two points. Consider the four-layer design in Figure 4(c) for example. The construction scheme for this eight-run MLD is illustrated in Figure 6. It is worth noting that we can also stop this construction process earlier to have only $n/2g$ layers for the design, with $2g$ points on each layer ($g = 1, 2, \ldots, n/2$). However, as will be seen later, MLD with $n/2$ layers possess the most desirable projection properties, and therefore in this chapter we only consider choosing $g = 1$ and having $n/2$ layers for a n-run design.

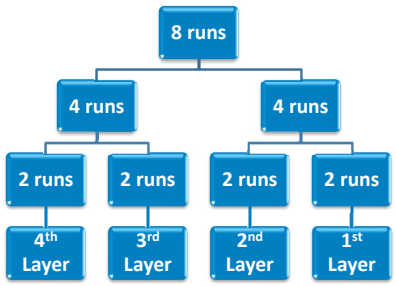

Figure 6: Construction scheme for a four-layer design.

In our construction strategy, the two points on each layer of MLD is a $2^{-(p-k-1)}$th fraction of the original 2^{p-k} design, which can be viewed as an individual $2^{p-(p-1)}$ design itself. According to Section 1.4, all these $2^{p-(p-1)}$ designs for each layer possess the same defining contrast subgroup (except for the signs), and their word length pattern can be denoted as $(A_1^*, A_2^*, \ldots, A_p^*)$. The next theorem shows the projection properties of MLDs.

Theorem 3. *If $A_1^* = 0$, the projection of MLD onto any one-dimensional subspace produces n distinct levels. If $A_1^* > 0$, when MLD is projected onto each of its dimensions, A_1^* of its factors have $n/2$ different levels while all the other $p - A_1^*$ factors have n distinct levels.*

Proof. In our method, MLD is constructed by allocating its n designs points evenly

into $n/2$ layers, with each layer containing exactly two points. This layered-structure guarantees that each factor of MLD has at least $n/2$ distinct levels, since there are $n/2$ different layers. On each layer, when the $2^{p-(p-1)}$ design has no word of length one in its defining contrast subgroup ($A_1^* = 0$), none of the factors are aliased with the positive/negative identity element and each factor has two different levels. Thus, the $n/2$ layers totaly project $\frac{n}{2} \times 2 = n$ levels onto each single dimension of the design. On the other hand, when on each layer the $2^{p-(p-1)}$ design has $A_1^* > 0$, then A_1^* out of p factors are fully aliased with the positive/negative identity (I or $-I$). In each layer, these A_1^* factors can only have one level while the other $p - A_1^*$ factors still take two different levels. Overall, when MLD is projected onto one-dimensional subspaces, A_1^* factors take $\frac{n}{2} \times 1$ levels and the other $p - A_1^*$ factors have $\frac{n}{2} \times 2$ different levels. $\diamondsuit$

We want to make two remarks on this projection property of MLD. First, since in MLD the points are split into layers by iteratively taking minimum aberration half-designs, the value of A_1^* has been minimized (less aberration). As a result, even when A_1^* turns out to be positive, it is usually a small number compared to the total dimension p. For example, the 16-run MLD in 8 factors has $A_1^* = 0$, and the 32-run MLD in 17 factors only has $A_1^* = 1$. Thus, in MLD the majority of factors are guaranteed to have n distinct levels, and only few, if any, factors would have $n/2$ levels. In the literature, it is well-known that LHDs, although not necessarily space-filling, possess the favorable feature that their one-dimensional projections always have n distinct levels. In this regard, the one-dimensional projection property of MLD is as good as LHD when $A_1^* = 0$. When $A_1^* > 0$, MLD becomes slightly worse, but in many cases, the average number of levels for factors in MLD is still close to that of LHD. Our second remark is, for a moderate sized design with $A_1^* > 0$, $n/2$ levels for design factors are usually large enough to capture the complex nonlinear functional relationships. Some other authors (such as Bingham et al. 2009) have also made similar arguments that although designs with many levels are desirable, it is not essential that the number of levels for each factor must be as large as the number of runs, as in the case of LHD. Therefore, when $A_1^* > 0$, factors with only $n/2$ levels would not be a serious disadvantage for MLD; instead, the overall space-filling

property of a design should be more critical in designing computer experiments than its one-dimensional projection properties.

The second question of this section is on how to choose the spacing between layers. Suppose the design region is scaled into $[-1, 1]^p$ and the base design points have been split into L layers. To achieve good space-filling properties, we artificially add an empty layer to the boundary of design region and study the spacing among $L + 1$ layers altogether. The most intuitive uniform spacing is to assign $i/(L + 1)$ scale for the ith layer, $i = 1, 2, \ldots, L + 1$. However, when L is large, this equal spacing makes points on innermost layer very close to each other, which drastically reduces the minimum distance between points. To avoid this problem, we deliberately leave a prescribed region $(-s, s)^p$ empty in the center of design space, and distribute the $L+1$ layers uniformly in the remaining area. With this scheme, the points are neither too close to the center nor to the boundary, but spread out uniformly in between, which makes MLD a good compromise between the maximin and minimax designs. Another pure geometric justification for leaving this small central region empty is that since the volume of design region outside is much larger than inside $(-s, s)^p$, it is reasonable to place more layers (points) to the outer area. To implement this strategy, we can apply $s+\frac{1-s}{L}(i-1), i = 1, 2, \ldots, L+1$ scales for each layer, where $s \in [0, 1]$ is the scale (or shrinkage parameter) for the innermost layer of MLD. The value of s determines the space-filling properties of MLD and needs to be chosen judiciously. The following lemma illustrates the optimal choices of s in two-dimensional cases.

Lemma 3. *For four-run MLD in two factors: (i) the optimal value of s with respect to minimax criterion is 0.4768 and the corresponding maximum distance is 0.73987; (ii) the optimal value of s with respect to maximin criterion is 1 and the corresponding minimum distance is 2.*

Part (i) of Lemma 3 can be shown by analytic geometry, and the proof is left in the appendix. Using similar methods, we can also show that, if the $(L + 1)$th empty layer were not artificially added to the boundary of design region, the optimal s would be 0.4142 and the maximum distance would increase from 0.73987 to 0.8284.

This justifies the addition of the empty layer since it indeed improves the space-filling property of MLD. Proof for part (ii) in Lemma 3 is trivial, since MLD with $s = 1$ degenerates to the single-layer 2^2 design, which is the maximin design.

Lemma 3 clearly manifests the disagreement in maximin and minimax criteria for this two-dimensional case. Obviously, the optimal value given by minimax criterion is more reasonable here, since the maximin value $s = 1$ only places points on the boundary. As discussed in section 1.2, neither of these two criteria can be sufficient when used alone, and the MLD should achieve a good compromise between them. For MLD in higher dimensions, the geometry becomes very complex and analytical solutions for optimal s become intractable. In next section, good values of s for general cases are searched by simulation, using optimal results found in two dimensions as starting values.

1.6 Numerical Studies

In this section, we simulate forty-eight representative designs in various dimensions and sizes, and study their space-filling properties for different choices of s. They include five 8-run designs ($3 \leq p \leq 7$), twelve 16-run designs ($4 \leq p \leq 15$), sixteen 32-run designs ($5 \leq p \leq 20$), and fifteen 64-run designs ($6 \leq p \leq 20$). In each case, the catalog of minimum aberration 2^{p-k} designs in Wu and Hamada (2009) is used as the source of base designs for MLDs. R codes for constructing MLDs (with any given s) are available from the authors upon request.

1.6.1 Optimal Value of s

Given a design $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ in the region $\mathcal{X}$, we let $d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ denote the Euclidean distance between its two points $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. The maximin criterion by definition is to maximize the minimum interpoint distance, measured by $\min_{1 \leq i < j \leq n} d(\boldsymbol{x}_i, \boldsymbol{x}_j)$. An extension to this definition proposed by Morris and Mitchell (1995) is to minimize the average interpoint distance of the design, given by

$$\phi_r = \Big(\frac{2}{n(n-1)} \sum_{1 \leq i < j \leq n} \frac{1}{d(\boldsymbol{x}_i, \boldsymbol{x}_j)^r} \Big)^{\frac{1}{r}}. \tag{4}$$

When r is sufficiently large, it can be shown that the ϕ_r criterion is equivalent to the maximin criterion. In our study, we choose a small value $r = 2$ and use ϕ_2 as a supplement to the minimum distance measure.

When it comes to the minimax criterion, we need to measure the maximum distance from any point in region $\mathcal{X}$ to design D, that is, the value of $\max_{\boldsymbol{x}\in\mathcal{X}} d(\boldsymbol{x}, D)$. However, except for a few special cases, this measure is very difficult to compute. A common strategy is to sample N points from the design region $\mathcal{X}$ and use distance from the furthest point to the design as an approximation. Since the sample size N is required to be extremely large, sophisticated optimal sampling methods are generally not applicable here. Among the few feasible choices, Latin hypercube sampling is usually considered superior to other easy methods such as simple random sampling, or sampling with regular grids. Thus, in our study we choose to explore the design region by $N = 1,000,000$ random Latin hypercube sample points, and the value of $\max_{1\le k\le N} d(\boldsymbol{x}_k, D)$ can be used as an approximation of the maximum distance measure.

By simulating MLD with various choices of $s \in [0, 1]$, we find that the maximin and minimax criteria are in substantial disagreement: maximizing the minimum distance tends to push layers towards the design boundary (e.g. larger s); while minimizing the maximum distance favors having more layers in the interior (e.g. moderate or smaller s). As discussed in Section 1.2, this contradiction is not unexpected for MLD, considering its points are always sparse in the design region ($n \le 2^p$). To seek a satisfactory compromise between these two criteria, the popularly used Maximin LHD (Mm LHD) is chosen as a benchmark for deciding the optimal value of s. There are basically two reasons for this choice. First, unlike the pure maximin or minimax designs, Mm LHDs does not focus on optimizing a single criterion. Instead, since Mm LHD is constrained to have n distinct projections in each dimension, it can be considered as a compromise between several criteria. Second, in computer experiments, Mm LHDs are the most widely used space-filling designs so far, which are expected to perform reasonably well in terms of both criteria. This makes them the most natural competitors for the new proposed designs. The pure maximin or minimax designs

(or even the minimax LHDs), on the other hand, are generally hard to construct and rarely used in practice. Therefore, we can select the optimal value of s by tuning MLDs to be superior to Mm LHDs with respect to both criteria.

In our study, Mm LHDs are generated by the commercial software JMP 8 (with 70 random starts for optimization). For a fair comparison, all designs are scaled into $(0,1)^p$ with no points on the boundary. As shown in Figures 7 and 9, MLDs with $s = 0.4, 0.45, 0.5$ have the closest performance to Mm LHDs in terms of the maximin and minimax criteria. For clarity of presentations, MLDs with other values of s are omitted from the plots. In Figure 7, we can see that MLDs with s=0.45 and 0.5 are obviously preferable to Mm LHDs in terms of the maximin criteria, while $s = 0.4$ appears to be worse. In addition, when their average interpoint distances ϕ_2 are compared in Figure 8, MLDs with all three choices of s are substantially more desirable than the Mm LHDs. When the maximum distance measure is evaluated in Figure 9, MLDs with $s = 0.45$ and 0.4 perform equally well with Mm LHDs. Their results are very close and comparable. MLDs with $s = 0.5$, on the other hand, are slightly worse than Mm LHDs for most cases (32 out of 48 cases). Since we want MLD to be better or as good as Mm LHD in terms of both measures, we recommend choosing 0.45 as a good tradeoff for s, which is also close to the optimal s value obtained analytically in the two-dimensional case in Lemma 3. Compared with the Mm LHDs, MLDs with $s = 0.45$ have a clear advantage in the minimum distance and also perform equally well in terms of the maximum distance.

1.6.2 Combined Criteria

We can also combine the minimum and maximum distances into a single measure to judge the overall desirability of a design. In this subsection, two intuitive combined criteria are introduced to further compare the proposed MLDs with Mm LHDs.

The first combined criterion is to maximize the ratio of the minimum distance measure to the maximum distance measure

$$\frac{\min_{1\le i<j\le n} d(\boldsymbol{x}_i, \boldsymbol{x}_j)}{\max_{1\le k\le N} d(\boldsymbol{x}_k, D)}. \tag{5}$$

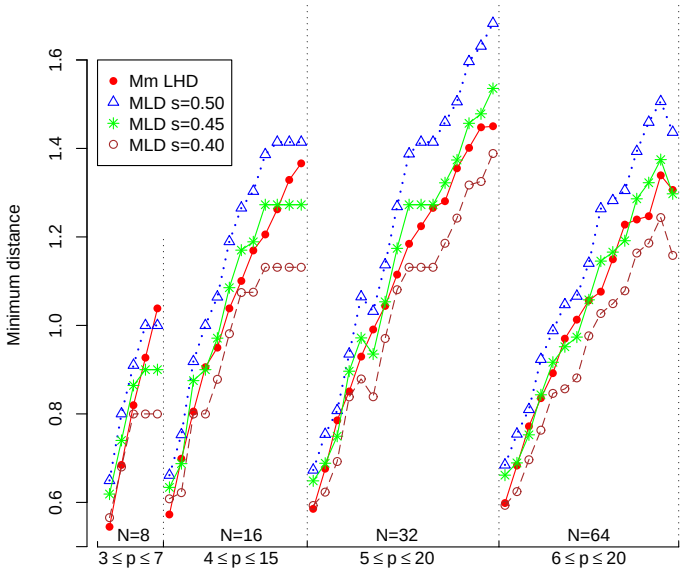

Figure 7: Minimum distances (larger-the-better).

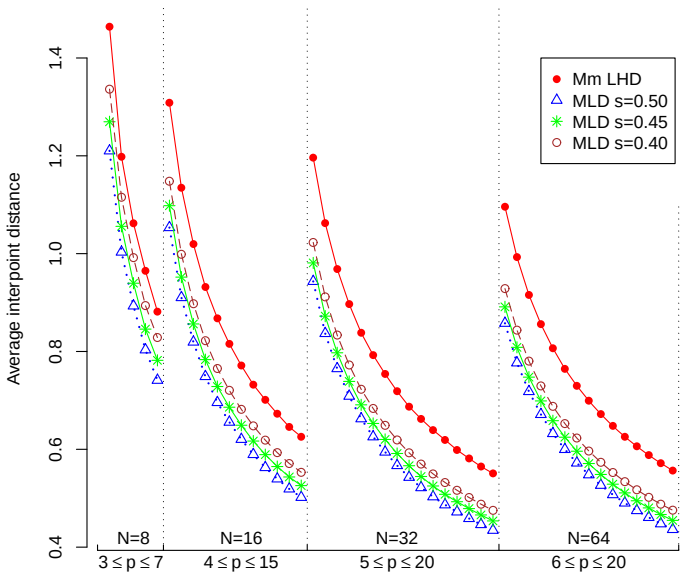

Figure 8: Average interpoint distances (smaller-the-better).

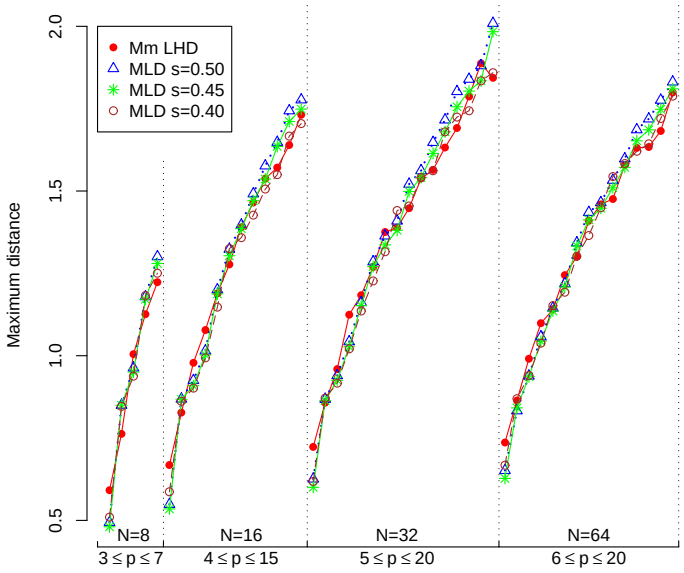

Figure 9: Maximum distances (smaller-the-better).

This is reasonable because the numerator and denominator of (5) are in the same scale.

Alternatively, similar to defining ϕ_r in (4) as an extension of the maximin criterion, we can also extend the minimax criterion to a counterpart, and combine it with ϕ_r to form a overall space-filling measure. Specifically, to determine the distance from an arbitrary point $\boldsymbol{x}_k \in \mathcal{X}$ to design D, instead of only considering distance from $\boldsymbol{x}_k$ to its nearest design point $\min_{\boldsymbol{x}_i \in D} d(\boldsymbol{x}_k, \boldsymbol{x}_i)$, we propose to define it as the average distance from $\boldsymbol{x}_k$ to all design points $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n\}$ in D

$$\rho_q(\boldsymbol{x}_k, D) = \Big(\frac{1}{n}\sum_{i=1}^{n} \frac{1}{d(\boldsymbol{x}_k, \boldsymbol{x}_i)^q}\Big)^{\frac{1}{q}}. \tag{6}$$

Note that this form is similar with (4), and the $\rho_q(\boldsymbol{x}_k, D)$ and ϕ_r are in the same scale. Since $\rho_q(\boldsymbol{x}_k, D) \to 1/\min_{\boldsymbol{x}_i \in D} d(\boldsymbol{x}_k, \boldsymbol{x}_i)$ as $q \to \infty$, the average distance in (6) can be considered as a generalization of the traditional $d(\boldsymbol{x}_k, D) = \min_{\boldsymbol{x}_i \in D} d(\boldsymbol{x}_k, \boldsymbol{x}_i)$. The form of $\rho_q(\boldsymbol{x}_k, D)$ also has a justification from the Inverse Distance Weighting (IDW) method (Shepard 1968), where the weighting function for each design point is

just of the form $w_i(\boldsymbol{x}) = 1/d(\boldsymbol{x}, \boldsymbol{x}_i)^q, i = 1, 2, \ldots, n$. By this definition, a reasonable criterion is to maximize $\min_{1 \le k \le N} \rho_q(\boldsymbol{x}_k, D)$. Combing this with ϕ_r, an overall space-filling measure analog to that in (5) can be developed as

$$\frac{\phi_r}{\min_{1 \le k \le N} \rho_q(\boldsymbol{x}_k, D)}. \tag{7}$$

Minimizing this measure can optimize the maximin and minimax criteria simultaneously. In our numerical study, we choose r and q to be 2, which is the commonly used value in defining weighting functions for IDW (Joseph and Kang 2011).

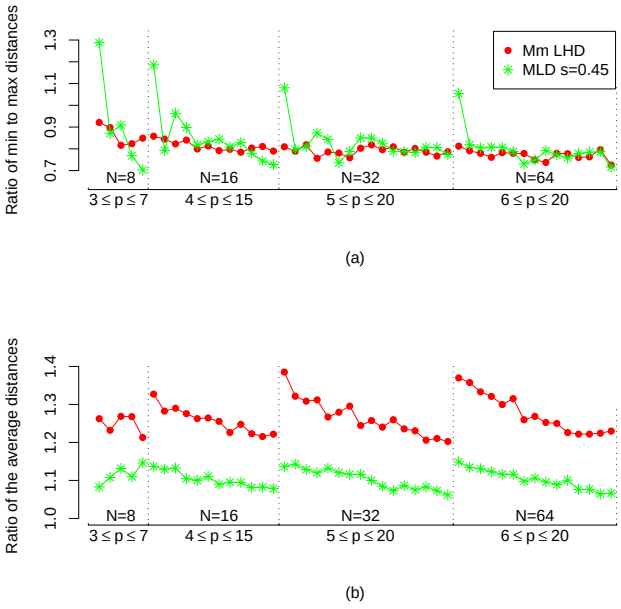

Figure 10: (a) Ratio of minimum to maximum distances (larger-the-better); (b) Ratio of the average distances (smaller-the-better).

In Figure 10(a), we can see that MLDs with $s = 0.45$ are slightly superior in terms of the ratio (5). When it comes to the combined measure (7), Figure 10(b) shows that MLDs are greatly preferable to Mm LHDs. These results clearly manifest that MLD with $s = 0.45$ is able to strike a desirable balance between the maximin and minimax designs and yield very appealing space-filling properties.

1.6.3 Saving on Computational Time

In addition to superior space-filling properties, it is also important to note that MLDs are much easier to construct than the optimal LHDs. For example, consider constructing moderate-sized designs in 14 factors. The commercial software JMP 8 (with 70 random starts for optimization) on a 2.66 GHz desktop takes 23 seconds to generate a 32-run Mm LHD, 3 minutes to generate a 64-run Mm LHD and 22.5 minutes to generate a 128-run Mm LHD; while our simple R code can construct each corresponding MLD within 3 or 4 seconds. For designs with larger number of factors and/or larger number of runs, the time saving becomes even more substantial.

1.7 Flexible Run Size

Previously, MLDs are constructed by splitting 2^{p-k} base design points into several layers. However, the run size for these base designs can only be a power of two, which limits their flexibility. In this section, we provide methods to extend MLDs to have more flexible number of runs.

For design in p factors, we first assume the run size to be even and within the range $n \in (2^{p-k-1}, 2^{p-k}), k = 0, 1, 2 \ldots$. The most natural way to fill up this gap is to add $n - 2^{p-k-1}$ extra points to a smaller MLD in 2^{p-k-1} runs. By geometry, it can be seen that in the design region $[-1, 1]^p$, the furthest points to MLD are among those $2^p - 2^{p-k-1}$ corner points that do not belong to the base design. Therefore, to add extra points, we can restrict our attention only to the missing fractions in 2^{p-k-1} base design. However, selecting an optimal subset of $n - 2^{p-k-1}$ points out of these $2^p - 2^{p-k-1}$ candidates still seems to be a very challenging task. Fortunately, this obstacle can be circumvented by utilizing the existing results on *optimal foldover plans* (Li and Lin 2003). Given a 2^{p-k-1} initial design, a popular follow-up strategy in physical experiments is to add a second fraction (i.e. the foldover design) by reversing signs of one or more columns of the initial design. Among the 2^p possible choices for this second fraction, the optimal foldover plan is defined to be the one such that the combined design has minimum aberration among all possible combined designs. Since minimum aberration designs possess favorable space-filling properties

as discussed in Section 1.3, for any MLD, the optimal foldover design for its base design naturally suggests the most ideal places to add extra points. In this way, the number of candidate points can be greatly reduced from $2^p - 2^{p-k-1}$ to 2^{p-k-1}, and we only need to select a desirable subset from these 2^{p-k-1} foldover points as extra points. Specifically, after obtaining a smaller MLD in 2^{p-k-1} runs as the *main layers*, we can select the additional $n - 2^{p-k-1}$ extra points with the following steps: (i) Find the optimal foldover plan for its base design according to the catalog in Li and Lin (2003). (ii) Obtain an optimal subset of this foldover design which consists the additional points for the design. This can be done by splitting the foldover design into optimal half-designs iteratively as in Section 1.4 and retaining the necessary parts each time. The resulting extra points can be arranged in $(n - 2^{p-k-1})/2$ layers, which constitute the *additional layers* for MLD. (iii) In the end, all main and additional layers are collected together and scaled as in Section 1.5, to form a design in $2^{p-k-1}/2 + (n - 2^{p-k-1})/2 = n/2$ layers.

Take the 24-run design in 9 factors for example. Since $24 \in [16, 32]$, we first split a 16-run 2^{9-5} base design into eight layers, and consider adding extra points from its optimal foldover plan (which can be obtained by reversing signs for factor 9). When this foldover design is split into two optimal half-designs, its first half-design comprises the 8 extra points that we need. Arranging these extra points in four layers and combining with the previous eight layers, we obtain the $24(= 16 + 8)$ run design in twelve layers. Similarly, if a 28-run design is needed, we can further split the second half-design into another two 4-run designs, and add points from one of them to form the combined design $(28 = 16 + 8 + 4)$.

Another restriction of MLD is that its maximum run size is 2^p, which tends to be small in very low dimensions, say $p \leq 4$. Although most applications in computer experiments are characterized by moderate or high dimensional inputs and small run sizes, in cases where more than 2^p runs are needed, the MLD can be augmented by adding points using a space-filling criterion. This can be done as follows: (i) construct a MLD in 2^p runs and fix these points; (ii) further add $n - 2^p$ points according to some optimal design criteria (such as maximin distance criterion). The optimization

of extra points in this step should also take into account all MLD points fixed in the previous step. As a special case, if grids are properly superimposed onto the design region, we can also add extra points by searching optimal LHD in $n - 2^p$ runs (optimized with respect to all n design points). Since MLDs are very easy to generate, by fixing 2^p MLD points in advance, considerable computational efforts can be saved for constructing good space-filling designs.

1.8 Conclusions

In this chapter, we presented a new type of space-filling designs for computer experiments. The proposed MLDs can be developed by splitting two-level full or fractional factorial designs into multiple layers. Due to their elegant structures, the construction of MLDs can make use of many available results in physical experiments and, as a result, the MLDs are very easy to generate. This is in contrast to most other popular space-filling designs, whose construction normally requires an intensive computer search. In addition, by properly choosing the spacing between layers, MLDs can strike a good balance between the maximin and minimax designs, and even outperform the Mm LHDs in terms of space filling.

Finally, we also want to note that the construction process for MLD can be easily adjusted when some prior knowledge about the response surface are available. For example, in some cases engineering domain knowledge may suggest that high order nonlinear effects in the model are probably not significant but high order interactions among factors are very likely to occur. In this situation, keeping n levels for each factor in the design is no longer desirable, and we may want to use fewer layers and moderately increase the redundancy of design points when they are projected onto each dimension. To accomplish this, the base design points can be split into fewer than $n/2$ layers. Then, each layer can contain more than two points, which helps in estimating some of the higher order interactions, but at the expense of some of the higher order nonlinear terms. For the optimal splitting of the points, understanding of the aliasing between factor interactions and nonlinear effects is essential, which we leave for future work.

1.9 Appendix: Proof of Lemma 3

Consider placing four design points P_1, P_2, P_3 and P_4 within the $[-1,1]^2$ region $ABCD$ shown in Figure 11. Denote the coordinates of these four points as $(s_1, -s_1)$, $(-s_1, s_1)$, $(-s_2, -s_2)$ and (s_2, s_2) respectively, where the first two points form the first layer and the last two constitute the second layer of MLD. These layers are constructed by adding new defining relation $I = -12$ or $I = 12$ to the 2^2 base design as discussed in Lemma 2. Given $s \in [0,1]$ as the scale for innermost layer of MLD, we have $s_1 = s$ and $s_2 = s + \frac{1-s}{2} \times 1 = \frac{s+1}{2}$.

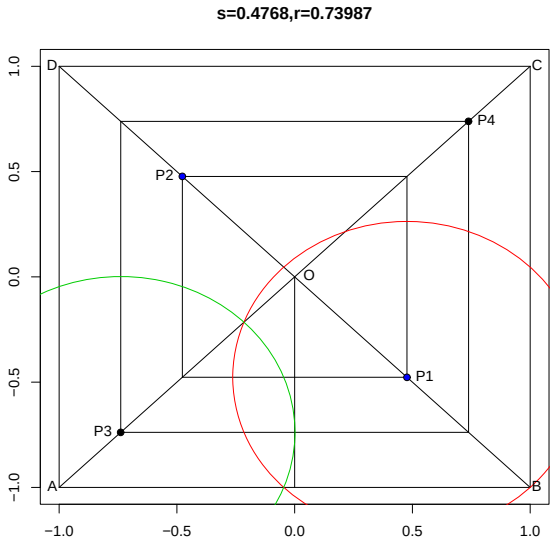

Figure 11: Optimal scaling for MLD in two dimensions.

We first consider part (i) of Lemma 3. If we draw four circles centered at P_1, P_2, P_3 and P_4 with equal radius r, the distance from any point inside the circle to its closest design point is less than or equal to r. For points outside these circles, however, their distances from design points are strictly larger than r. As we gradually increase the radius r, more and more points get covered by those circles. Among all points in $[-1,1]^2$ region, the last one to be covered is furthest away from the design, and we want to minimize this maximum distance. Since the four triangle regions

OAB, OBC, OCD and OAD in $ABCD$ are symmetric to each other, points in each of them are reflections of the others. Therefore we can restrict our attention only to the triangle region OAB and study the maximum distance from points within this region to its design points P_1 and P_3.

Two circles centered at P_1 and P_3 are drawn with equal radius r in Figure 11. As we gradually increase r, it can be seen that the last point to be covered in region OAB may exist in three possible locations: (a) on segment OA where the two circles can intersect; (b) on segment AB where the two circles can intersect; (c) the corner point B at (1,-1). The necessary radiuses to cover each of these candidate points are calculated as follows.

(a) Denote the coordinates of candidate point on OA as (x_1, x_1). Since the two circles intercept at this point, the distances from it to both center points P_1 and P_3 are equal. Thus, we have: $(x_1 - s)^2 + (x_1 + s)^2 = (x_1 + \frac{s+1}{2})^2 + (x_1 + \frac{s+1}{2})^2$, which yields $x_1 = \frac{(3s+1)(s-1)}{4(s+1)}$. The corresponding radius to cover this point is $r_1(s) = \sqrt{2x_1^2 + 2s^2} = \sqrt{\frac{(3s+1)^2(s-1)^2}{8(s+1)^2} + 2s^2}$.

(b) Denote the coordinates of candidate point on AB as $(x_2, -1)$. Since the two circles intercept at this point, the distances from it to both center points P_1 and P_3 are equal. Thus, we have: $(x_2 - s)^2 + (1 - s)^2 = (x_2 + \frac{s+1}{2})^2 + (1 - \frac{s+1}{2})^2$, which yields $x_2 = \frac{(3s-1)(s-1)}{2(3s+1)}$. The corresponding radius to cover this point is $r_2(s) = \sqrt{(x_2 - s)^2 + (1 - s)^2} = \sqrt{(\frac{3s^2+6s-1}{6s+2})^2 + (1 - s)^2}$.

(c) The radius to cover point B at (1,-1) by its closest design point P_1 is: $r_3(s) = \sqrt{(s-1)^2 + (1-s)^2} = \sqrt{2} - \sqrt{2}s$.

Taking the maximum of above radiuses, we can obtain the maximum distance from any point in OAB to the design: $r(s) = \max(r_1(s), r_2(s), r_3(s))$. The optimal value of s with respect to the minimax criterion is defined as:

$$r(s_{mM}) = \min_s r(s) = \min_s \max(r_1(s), r_2(s), r_3(s)),$$

which leads to $s_{mM} = 0.4768$ and $r(s_{mM}) = 0.73987$. The plot of $r_1(s), r_2(s), r_3(s)$ with respect to $s \in [0, 1]$ is shown in Figure 12. Note that when s is optimal, radiuses

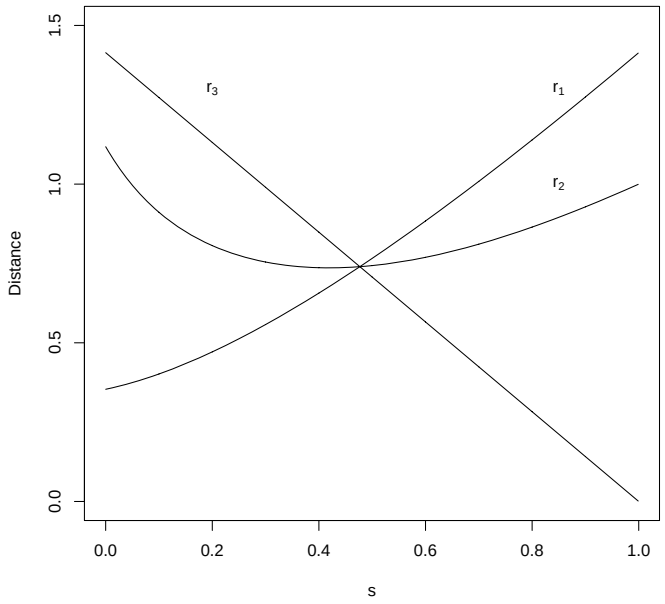

Figure 12: Optimal value of s with respect to minimax criterion.

r_1, r_2, r_3 happen to be equally large and all three candidate points in (a),(b),(c) attain the maximum distance $r(s_{mM})$. This optimal scaling scheme is also illustrated in Figure 11. For other s values, however, the values for three radiuses are different and only one candidate point can stand out to be furthest away from the design.

Part (ii) of Lemma 3 is trivial, since when $s_{Mm} = 1$, both s_1 and s_2 equal to 1 and MLD degenerates to a single-layer 2^2 design, which is the maximin design in general case.

CHAPTER II

COMPOSITE GAUSSIAN PROCESS MODELS FOR EMULATING EXPENSIVE FUNCTIONS

2.1 Introduction

The modern era witnesses the prosperity of computer experiments, which play a critical role in many fields of technological development where the traditional physical experiments are infeasible or unaffordable to conduct. By developing sophisticated computer simulators, people are able to evaluate, optimize and test complex engineering systems even before building expensive prototypes. The computer simulations are usually deterministic (no random error), yield highly nonlinear response surfaces, and are very time-consuming to run. To facilitate the analysis and optimization of the underlying system, surrogate models (or emulators) are often fitted to approximate the unknown simulated surface based on a finite number of evaluations (Sacks, Welch, Mitchell and Wynn 1989). Santner, Williams and Notz (2003) and Fang, Li and Sudjianto (2006) provide detailed reviews on the related topics.

In computer experiments, the stationary Gaussian process (GP) model is popularly used for approximating computationally expensive simulations. Its framework is built on modeling the computer outputs $Y(\boldsymbol{x}), \boldsymbol{x} \in \mathbb{R}^p$ as a realization of a stationary GP with constant mean μ and covariance function $\sigma^2 \mathrm{cov}(Y(\boldsymbol{x}+\boldsymbol{h}), Y(\boldsymbol{x})) = \sigma^2 R(\boldsymbol{h})$, where the correlation $R(\boldsymbol{h})$ is a positive semidefinite function with $R(\boldsymbol{0}) = 1$ and $R(-\boldsymbol{h}) = R(\boldsymbol{h})$. When the above assumptions are satisfied, the corresponding predictor can be shown to be a *best linear unbiased predictor* (BLUP), in the sense that it minimizes the mean squared prediction error. Nevertheless, many studies in the literature have pointed out that the artificial assumption of second-order stationarity for the GP model are more for theoretical convenience rather than for representing reality, and they can be easily challenged in practice. If these assumptions deviate

from the truth, the predictor is no longer optimal, and sometimes can even be problematic (see the discussions, for example, in Joseph 2006, Xiong et al. 2007, Gramacy and Lee 2011).

When the constant mean assumption for the GP model is violated, a frequently observed consequence is that the predictor tends to revert to the global mean, especially at locations far from design points. Consider a simple example from Xiong et al. (2007). Suppose the true function is $y(x) = \sin(30(x-0.9)^4)\cos(2(x-0.9)) + (x-0.9)/2$ and we choose 17 unequally spaced points from [0,1] to evaluate the function. The function and design points are illustrated in Figure 13. Obviously, the mean of this function in region $x \in [0, 0.4]$ is much smaller than the mean in region $x \in [0.4, 1]$. When the data are fitted with a stationary GP model with a Gaussian correlation function, a constant mean for the whole region is estimated as -0.147 by maximizing the likelihood function (Santner et al. 2003, page 66), and the corresponding predictor along with this mean value are shown in Figure 13. Clearly, the fit in region $x \in [0.4, 1]$ is not good, since the prediction is pulled down to the global mean.

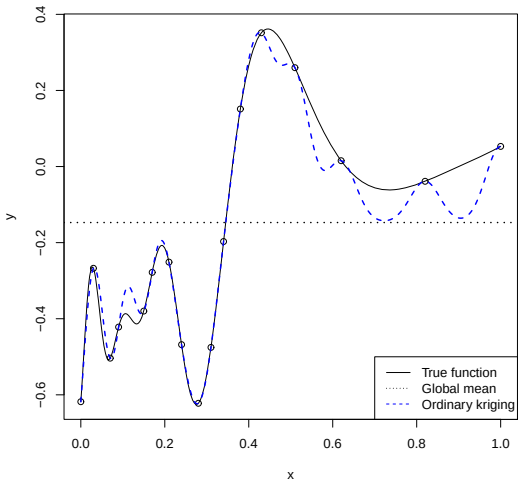

Figure 13: Plot of function $y(x) = \sin(30(x-0.9)^4)\cos(2(x-0.9)) + (x-0.9)/2$, the global mean and the ordinary kriging predictor.

Just as a non-constant global trend can be quite common in engineering systems, the variability of simulated outputs can also change dramatically throughout the

design region. Still consider the simple case in Figure 13 for example: the roughness of the one-dimensional function in region $x \in [0, 0.4]$ is much larger than in region $x \in [0.4, 1]$. For the GP model assuming a constant variance for the whole input region, the variance estimate for region $x \in [0.4, 1]$ tends to be inflated by averaging with that of the other part, which further contributes to the erratic prediction in this region. It is expected that as we increase the simulation sample size, the above problem can be mitigated. However, since most typical applications of computer experiments involve high dimensional inputs, the data points always tend to be sparse in the design region and it is almost impossible to avoid such kind of gaps in practice.

In this chapter, we propose a more accurate modeling approach by incorporating a flexible global trend and a variance model into the GP model. The proposed predictor has an intuitive structure and can be efficiently estimated in a single stage. Not only can the new predictor mitigate the problems discussed above, it also enjoys several additional advantages such as better numerical stability, robustness to sparse design and improved prediction intervals.

The chapter is organized as follows. Section 2.2 introduces the notation and existing work. Section 2.3 presents the new predictor and shows its interesting connections with some existing methods. In Section 2.4 we discuss how to estimate the unknown parameters by maximum likelihood. Several properties of the new predictor are studied in Section 2.5, and in Section 2.6 we use several examples to demonstrate the advantages of the new method. Some final concluding remarks are given in Section 2.7.

2.2 Notation and Existing Work

In the computer experiments literature, the GP model is also often referred to as the *kriging* model (Currin et al. 1991), and these two terms are used interchangeably in this chapter. Suppose we have run the simulations under n different input settings $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset \mathbb{R}^p$. Denote the corresponding computer outputs as $\boldsymbol{y} = (y_1, \ldots, y_n)^\top$. A stationary GP model, called *ordinary kriging*, can be formally stated as

$$Y(\boldsymbol{x}) = \mu + Z(\boldsymbol{x}), \tag{8}$$

where $Z(\boldsymbol{x}) \sim GP(0, \sigma^2 R(\cdot))$. The ordinary kriging predictor at an input location $\boldsymbol{x}$ is given by

$$\hat{y}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{R}^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}), \tag{9}$$

where $\boldsymbol{r}(\boldsymbol{x}) = (R(\boldsymbol{x} - \boldsymbol{x}_1), \ldots, R(\boldsymbol{x} - \boldsymbol{x}_n))^\top$, $\boldsymbol{R}$ is an $n \times n$ correlation matrix with the $(ij)^{\text{th}}$ element $R(\boldsymbol{x}_i - \boldsymbol{x}_j)$, $\mathbf{1}$ is a n-dimensional vector with all elements 1, and $\hat{\mu} = (\mathbf{1}^\top \boldsymbol{R}^{-1}\mathbf{1})^{-1}(\mathbf{1}^\top \boldsymbol{R}^{-1}\boldsymbol{y})$.

To remedy the the predictor's reversion to mean problem as discussed in the previous section, a common strategy is to relax the constant mean μ in ordinary kriging with a *global trend* $\mu(\boldsymbol{x})$, and modify the model in (8) as

$$Y(\boldsymbol{x}) = \mu(\boldsymbol{x}) + Z(\boldsymbol{x}). \tag{10}$$

If the global trend is comprised of some prescribed polynomial models $\mu(\boldsymbol{x}) = \boldsymbol{f}^\top(\boldsymbol{x})\boldsymbol{\beta}$, where $\boldsymbol{f}(\boldsymbol{x}) = (1, f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^\top$ are known functions and $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_m)^\top$ are unknown parameters, the model in (10) is called *universal kriging*. Define a $n \times (m+1)$ matrix $\boldsymbol{F} = (\boldsymbol{f}(\boldsymbol{x}_1), \ldots, \boldsymbol{f}(\boldsymbol{x}_n))^\top$, and the corresponding optimal predictor under model (10) can be derived as

$$\hat{y}(\boldsymbol{x}) = \boldsymbol{f}^\top(\boldsymbol{x})\hat{\boldsymbol{\beta}} + \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{R}^{-1}(\boldsymbol{y} - \boldsymbol{F}\hat{\boldsymbol{\beta}}), \tag{11}$$

where $\hat{\boldsymbol{\beta}} = (\boldsymbol{F}^\top \boldsymbol{R}^{-1}\boldsymbol{F})^{-1}(\boldsymbol{F}^\top \boldsymbol{R}^{-1}\boldsymbol{y})$. If $\mu(\boldsymbol{x}) = \boldsymbol{f}^\top(\boldsymbol{x})\boldsymbol{\beta}$ is close to the true global trend, then clearly this approach can give much better prediction than that of (9). However, in practice the correct functional form $\boldsymbol{f}(\boldsymbol{x}) = (1, f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x}))^\top$ is rarely known, and a wrongly specified trend in universal kriging can make the prediction even worse. For this reason, Welch et al. (1992) suggested using ordinary kriging instead of universal kriging. Another practical approach, called *blind kriging*, is to relax the assumption that the $f_i(\boldsymbol{x})$'s are known and select them from a candidate set of functions using a variable selection technique (Joseph, Hung and Sudjianto 2008). Although this strategy usually leads to better fit, performing the variable selection while interacting with the second stage GP model is a non-trivial task. Considerable computational efforts are needed to properly divide up the total variation between the polynomial trend and the GP model. In addition, in some cases, polynomial models may not be adequate to fit the complex global trend well.

Generalizing the GP model for non-stationary variance is an even more challenging task. None of the above remedies for the non-stationary mean can in any sense alleviate the constant variance restriction, and most studies in the literature focus on deriving complex non-stationary covariance functions such as by spatial deformations or kernel convolution approaches (for example, see Sampson and Guttorp 1992, Higdon, Swall, and Kern 1999, Schmidt and O'Hagan 2003, Paciorek and Schervish 2006, Anderes and Stein 2008). However, those structures may easily get overparameterized in high dimensions and become computationally intractable to fit. In addition, many of them also require multiple observations, which is not applicable to the single set of outputs from computer experiments. Some other work includes Xiong et al. (2007), which adopts a non-linear mapping approach based on a parameterized density function to incorporate the non-stationary covariance structure. Gramacy and Lee (2008) utilize the Bayesian treed structure to implement a non-stationary GP model. However, by dividing the design space into subregions, the treed GP model may lose efficiency since the prediction is only based on local information, and its response can also be discontinuous across subregions. In the next section, we propose to solve the non-stationarity problem via a different approach. We show that the flexible mean and variance models can be incorporated into GP by using the *composite Gaussian process* (CGP) models.

2.3 Composite Gaussian Process Models

For clarity, in this section we develop the new method in two steps. First, a predictor that intrinsically incorporates a flexible mean model is presented, and then we further augment it with a variance model to simultaneously handle the change of variability in the response.

2.3.1 Improving the Mean Model

The universal kriging (or blind kriging) in (10) contains a polynomial mean model $\mu(\boldsymbol{x})$ as the global trend and a kriging model $Z(\boldsymbol{x})$ for local adjustments. To avoid the awkward variable selections in $\mu(\boldsymbol{x})$ and also make the mean model more flexible,

we propose to use another GP to model the $\mu(\boldsymbol{x})$ as in the following form

$$\begin{aligned} Y(\boldsymbol{x}) &= Z_{global}(\boldsymbol{x}) + Z_{local}(\boldsymbol{x}), \\ Z_{global}(\boldsymbol{x}) &\sim GP(\mu, \tau^2 g(\cdot)), \\ Z_{local}(\boldsymbol{x}) &\sim GP(0, \sigma^2 l(\cdot)). \end{aligned} \tag{12}$$

Here the two GPs $Z_{global}(\boldsymbol{x})$ and $Z_{local}(\boldsymbol{x})$ are stationary and independent of each other. The first GP with variance τ^2 and correlation structure $g(\cdot)$ is required to be *smoother* to capture the global trend while the second GP with variance σ^2 and correlation $l(\cdot)$ is for local adjustments. Just as the universal kriging generalizes the ordinary kriging by adding a polynomial mean model $\mu(\boldsymbol{x})$, the new model in (12) can be viewed as a further extension which adopts a more sophisticated GP for global trend modeling. It is interesting to note that, the *linear model of regionalization* in geostatistics (Wackernagel 2003, Chapter 14) also employs a similar structure to model regionalized phenomena in geological data, but its final model form and estimation strategies are quite different from our approach.

Under the new assumptions in (12), the optimal predictor is easy to derive. Since the sum of two independent GPs is still a GP, we can equivalently express (12) as $Y(\boldsymbol{x}) \sim GP(\mu, \tau^2 g(\cdot) + \sigma^2 l(\cdot))$. Similar to ordinary kriging, the best linear unbiased predictor under the assumptions in (12) can be written as

$$\hat{y}(\boldsymbol{x}) = \hat{\mu} + (\boldsymbol{g}(\boldsymbol{x}) + \lambda \boldsymbol{l}(\boldsymbol{x}))^\top (\boldsymbol{G} + \lambda \boldsymbol{L})^{-1} (\boldsymbol{y} - \hat{\mu}\mathbf{1}) \tag{13}$$

where $\lambda = \sigma^2/\tau^2$ ($\lambda \in [0, 1]$) is the ratio of variances, $\boldsymbol{g}(\boldsymbol{x}) = (g(\boldsymbol{x} - \boldsymbol{x}_1), \ldots, g(\boldsymbol{x} - \boldsymbol{x}_n))^\top$, $\boldsymbol{l}(\boldsymbol{x}) = (l(\boldsymbol{x} - \boldsymbol{x}_1), \ldots, l(\boldsymbol{x} - \boldsymbol{x}_n))^\top$, $\boldsymbol{G}$ and $\boldsymbol{L}$ are two $n \times n$ correlation matrices with the $(ij)^{\text{th}}$ element $g(\boldsymbol{x}_i - \boldsymbol{x}_j)$ and $l(\boldsymbol{x}_i - \boldsymbol{x}_j)$ respectively, and $\hat{\mu} = (\mathbf{1}^\top (\boldsymbol{G} + \lambda \boldsymbol{L})^{-1} \mathbf{1})^{-1} \mathbf{1}^\top (\boldsymbol{G} + \lambda \boldsymbol{L})^{-1} \boldsymbol{y}$. Here the variance ratio λ is restricted to $[0, 1]$ because we expect the global trend to capture most of the variation in the response surface than the local process.

Although many possible correlation structures are available for $g(\cdot)$ and $l(\cdot)$, throughout this chapter we follow the standard choice in computer experiments and

specify them using the *Gaussian correlation functions*:

$$g(\boldsymbol{h}|\boldsymbol{\theta}) = \exp(-\sum_{j=1}^{p} \theta_j h_j^2), \qquad l(\boldsymbol{h}|\boldsymbol{\alpha}) = \exp(-\sum_{j=1}^{p} \alpha_j h_j^2), \tag{14}$$

where $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ are unknown correlation parameters satisfying $\mathbf{0} \leq \boldsymbol{\theta} \leq \boldsymbol{\alpha}^l$ and $\boldsymbol{\alpha}^l \leq \boldsymbol{\alpha}$. The bounds $\boldsymbol{\alpha}^l$ are usually set to be moderately large, which ensures that the component $Z_{global}(\boldsymbol{x})$ is indeed smoother than $Z_{local}(\boldsymbol{x})$ in the fitted model.

The new predictor in (13) is still an interpolator, since $\hat{y}(\boldsymbol{x}_i) = \hat{\mu} + \boldsymbol{e}_i^\top(\boldsymbol{y} - \hat{\mu}\mathbf{1}) = y_i$ for $i = 1, \ldots, n$, where $\boldsymbol{e}_i$ is a unit vector with a 1 at its ith position. It can also be seen that when $\lambda = 0$ (i.e. $\sigma^2 = 0$), the new model reduces to ordinary kriging. When $\lambda \in (0, 1]$, the predictor in (13) can be written out as the sum of a global predictor and a local predictor

$$\hat{y}(\boldsymbol{x}) = \hat{y}_{global}(\boldsymbol{x}) + \hat{y}_{local}(\boldsymbol{x}), \tag{15}$$

$$\hat{y}_{global}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{g}^\top(\boldsymbol{x})(\boldsymbol{G} + \lambda\boldsymbol{L})^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}), \tag{16}$$

$$\hat{y}_{local}(\boldsymbol{x}) = \lambda\boldsymbol{l}^\top(\boldsymbol{x})(\boldsymbol{G} + \lambda\boldsymbol{L})^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}). \tag{17}$$

It is important to note that, since the lower bounds for $\boldsymbol{\alpha}$ in (14) are usually set to be moderately large, the off-diagonal elements in $\boldsymbol{L}$ are closer to zero. Particularly, we can obtain $\boldsymbol{L} \to \boldsymbol{I}$ when $\boldsymbol{\alpha}$ take very large values. This immediately suggests two interesting properties for the CGP model. First, its global trend predictor $\hat{y}_{global}(\boldsymbol{x})$ in (16) resembles a kriging predictor with nugget effect as $\boldsymbol{L} \to \boldsymbol{I}$. When $\lambda > 0$, this nugget predictor is smooth but non-interpolating, and is commonly used in spatial statistics for modeling observational data with noise (Cressie 1991). Secondly, since $\boldsymbol{L} \approx \boldsymbol{I}$, the λ in $(\boldsymbol{G} + \lambda\boldsymbol{L})$ is mainly added to the diagonal elements. This makes $(\boldsymbol{G} + \lambda\boldsymbol{L})$ resistent to become ill-conditioned and the computation of $(\boldsymbol{G} + \lambda\boldsymbol{L})^{-1}$ in CGP can be numerically very stable. These two properties are elaborated in detail in Section 2.5.

2.3.2 Improving Both the Mean and Variance Models

To further relax the constant variance restriction, we introduce a variance model $\sigma^2(\boldsymbol{x})$ into (12) as follows

$$\begin{aligned} Y(\boldsymbol{x}) &= Z_{global}(\boldsymbol{x}) + \sigma(\boldsymbol{x}) Z_{local}(\boldsymbol{x}), \\ Z_{global}(\boldsymbol{x}) &\sim GP(\mu, \tau^2 g(\cdot)), \\ Z_{local}(\boldsymbol{x}) &\sim GP(0, l(\cdot)). \end{aligned} \tag{18}$$

The $Z_{global}(\boldsymbol{x})$ above remains the same as in (12), since the global trend is smooth and can reasonably be assumed to be stationary. After subtracting $Z_{global}(\boldsymbol{x})$ from the response, the second process is augmented with a variance model to quantify the change of local variability such that $\sigma(\boldsymbol{x}) Z_{local}(\boldsymbol{x}) \sim GP(0, \sigma^2(\boldsymbol{x}) l(\cdot))$. Overall, the model form in (18) is equivalent to assuming that the response $Y(\boldsymbol{x}) \sim GP(\mu, \tau^2 g(\cdot) + \sigma^2(\boldsymbol{x}) l(\cdot))$.

Without loss of generality, suppose the variance model can be expressed as $\sigma^2(\boldsymbol{x}) = \sigma^2 v(\boldsymbol{x})$, where σ^2 is an unknown variance constant and $v(\boldsymbol{x})$ is the standardized volatility function which fluctuates around the unit value. In the following discussion, we first assume that $v(\boldsymbol{x})$ is known, and denote $\boldsymbol{\Sigma} = diag\{v(\boldsymbol{x}_1), \ldots, v(\boldsymbol{x}_n)\}$ to represent the standardized local variances at each of the design points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. An efficient strategy for obtaining the $v(\boldsymbol{x})$ function is presented at the end of this section.

The model assumptions in (18) suggest that $y(\boldsymbol{x})$ and $\boldsymbol{y} = (y_1, \ldots, y_n)^\top$ have the multivariate normal distribution

$$\begin{pmatrix} y(\boldsymbol{x}) \\ \boldsymbol{y} \end{pmatrix} \tag{19}$$
$$\sim N_{1+n} \left[\begin{pmatrix} \mu \\ \mu \mathbf{1} \end{pmatrix}, \begin{pmatrix} \tau^2 + \sigma^2 v(\boldsymbol{x}) & (\tau^2 \boldsymbol{g}(\boldsymbol{x}) + \sigma^2 v^{1/2}(\boldsymbol{x}) \boldsymbol{\Sigma}^{1/2} \boldsymbol{l}(\boldsymbol{x}))^\top \\ \tau^2 \boldsymbol{g}(\boldsymbol{x}) + \sigma^2 v^{1/2}(\boldsymbol{x}) \boldsymbol{\Sigma}^{1/2} \boldsymbol{l}(\boldsymbol{x}) & \tau^2 \boldsymbol{G} + \sigma^2 \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2} \end{pmatrix} \right].$$

The best linear unbiased predictor under these assumptions can be derived as

$$\begin{aligned} \hat{y}(\boldsymbol{x}) =& \hat{\mu} + (\tau^2 \boldsymbol{g}(\boldsymbol{x}) + \sigma^2 v^{1/2}(\boldsymbol{x}) \boldsymbol{\Sigma}^{1/2} \boldsymbol{l}(\boldsymbol{x}))^\top (\tau^2 \boldsymbol{G} + \sigma^2 \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2})^{-1} (\boldsymbol{y} - \hat{\mu} \mathbf{1}) \\ =& \hat{\mu} + (\boldsymbol{g}(\boldsymbol{x}) + \lambda v^{1/2}(\boldsymbol{x}) \boldsymbol{\Sigma}^{1/2} \boldsymbol{l}(\boldsymbol{x}))^\top (\boldsymbol{G} + \lambda \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2})^{-1} (\boldsymbol{y} - \hat{\mu} \mathbf{1}), \end{aligned} \tag{20}$$

where $\lambda = \sigma^2/\tau^2$ ($\lambda \in [0,1]$), $\hat{\mu} = (\mathbf{1}^\top(\boldsymbol{G}+\lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}\mathbf{1})^{-1}\mathbf{1}^\top(\boldsymbol{G}+\lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}\boldsymbol{y}$ and all the other notations remain the same as in (13). Note that after defining the ratio λ, the unknown σ^2 is no longer needed for prediction, because the predictor depends on the variance model $\sigma^2(\boldsymbol{x})$ only through λ and $v(\boldsymbol{x})$. The predictor includes (13) as a special case when the local volatility model $v(\boldsymbol{x})$ degenerates to a constant function. The predictor can also interpolate all the data points since $(\boldsymbol{g}(\boldsymbol{x}_i)+\lambda v^{1/2}(\boldsymbol{x}_i)\boldsymbol{\Sigma}^{1/2}\boldsymbol{l}(\boldsymbol{x}_i))^\top(\boldsymbol{G}+\lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1} = \boldsymbol{e}_i^\top$ and $\hat{y}(\boldsymbol{x}_i) = \hat{\mu}+\boldsymbol{e}_i^\top(\boldsymbol{y}-\hat{\mu}\mathbf{1}) = y_i$ for $i = 1,\ldots,n$. By decomposing the predictor (20) into two parts

$$\hat{y}(\boldsymbol{x}) = \hat{y}_{global}(\boldsymbol{x}) + \hat{y}_{local}(\boldsymbol{x}), \tag{21}$$

$$\hat{y}_{global}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{g}^\top(\boldsymbol{x})(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}), \tag{22}$$

$$\hat{y}_{local}(\boldsymbol{x}) = \lambda v^{1/2}(\boldsymbol{x})\boldsymbol{l}^\top(\boldsymbol{x})\boldsymbol{\Sigma}^{1/2}(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}), \tag{23}$$

we can see that the global trend $\hat{y}_{global}(\boldsymbol{x})$ in (22) reduces to a *stochastic kriging* predictor (Ankenman, Nelson and Staum 2010) when $\boldsymbol{L} \to \boldsymbol{I}$. Different from the nugget predictor in (16) where a universal term λ is used for adjusting the global trend throughout the whole region, the amount of shrinkage at each data point in (22) is proportional to the value of $\lambda v(\boldsymbol{x}_i)$. This *localized adjustment* scheme is advantageous in making the global trend smoother and more stable, since it is less affected by the data points with large variability.

The above predictor form is derived based on $Y(\boldsymbol{x}) \sim GP(\mu, \tau^2 g(\cdot) + \sigma^2(\boldsymbol{x})l(\cdot))$, which unifies the modeling assumptions (18) in a *single stage*. As a result, the new method can also be viewed as extending the kriging model with a non-stationary covariance structure $\tau^2 g(\cdot) + \sigma^2(\boldsymbol{x})l(\cdot)$. Different from this, another strategy to fulfill the new assumptions in (18) is to develop the global and local models *sequentially*: (i) Fit a global trend model as in (22) using the likelihood method. (ii) Obtain its residuals $\boldsymbol{s} = (\boldsymbol{y} - \hat{\boldsymbol{y}}_{global})$, where $\hat{\boldsymbol{y}}_{global} = (\hat{y}_{global}(\boldsymbol{x}_1),\ldots,\hat{y}_{global}(\boldsymbol{x}_n))^\top$. If the estimated global trend interpolates all the data points ($\hat{\lambda} = 0$), we have $\boldsymbol{s} = \mathbf{0}$ and in this case the CGP just degenerates to a traditional single GP model. (iii) If $\boldsymbol{s} \neq \mathbf{0}$, standardize the residuals to achieve variance homogeneity $\boldsymbol{s}^* = \boldsymbol{\Sigma}^{-1/2}\boldsymbol{s}$. (iv) Adjust the global trend by interpolating the standardized residuals via a simple kriging model

$\hat{y}_{adj}(\boldsymbol{x}) = \boldsymbol{l}^\top(\boldsymbol{x})\boldsymbol{L}^{-1}\boldsymbol{s}^*$. In this way, we can form a sequential predictor as

$$\hat{y}_{seq}(\boldsymbol{x}) = \hat{y}_{global}(\boldsymbol{x}) + v^{1/2}(\boldsymbol{x})\hat{y}_{adj}(\boldsymbol{x}) = \hat{y}_{global}(\boldsymbol{x}) + v^{1/2}(\boldsymbol{x})\boldsymbol{l}^\top(\boldsymbol{x})\boldsymbol{L}^{-1}\boldsymbol{s}^*. \tag{24}$$

It is of natural interest to ask whether this sequential predictor would make any difference from the single-stage predictor (20), and the following theorem establishes their connections.

Theorem 4. *Given the same parameter values, the single-stage predictor (20) and the sequential predictor (24) are equivalent.*

Proof of the theorem is left in the Appendix. Despite this equivalent model form, we want to emphasize that the single-stage fitting strategy is superior to the sequential one in parameter estimation. This is because all parameters in the single-stage predictor (20) can be optimized simultaneously, which takes into account the interactions between global and local models and automatically balances their effects. In contrast to this global optimization, the sequential fitting approach estimates the parameters in two separate steps, and each of them can at most achieve local optimality. Generally, the global trend is hard to identify correctly without considering the effects of the second stage model, and in many cases the performance of the final prediction can be quite sensitive to this "global-local tradeoff". As a result, in this chapter we only consider the single-stage modeling framework, and this is also a major advantage for the proposed method over other multi-step strategies such as blind kriging.

In the rest of this section, we present how to obtain the $v(\boldsymbol{x})$ function, which is required for the CGP predictor. As shown in (21), the CGP model can be decomposed into a global and a local component, and this structure provides us a convenient way to assess the change of local volatility. For a given global trend (22) (initially we can set $\boldsymbol{\Sigma} = \boldsymbol{I}$), its squared residuals $\boldsymbol{s}^2 = (s_1^2, \ldots, s_n^2)^\top$ are natural measures of the local volatility, which can be used as the bases to build the $v(\boldsymbol{x})$ function. Based on $\boldsymbol{s}^2$, we propose an intuitive *Gaussian kernel regression model* for $v(\boldsymbol{x})$ as:

$$v(\boldsymbol{x}) = \frac{\boldsymbol{g}_b^\top(\boldsymbol{x})\boldsymbol{s}^2}{\boldsymbol{g}_b^\top(\boldsymbol{x})\boldsymbol{1}}, \tag{25}$$

where $\boldsymbol{g}_b(\boldsymbol{x}) = (g_b(\boldsymbol{x}-\boldsymbol{x}_1),\ldots,g_b(\boldsymbol{x}-\boldsymbol{x}_n))^\top$ with $g_b(\boldsymbol{h}|\boldsymbol{\theta},b) = \exp(-b\sum_{j=1}^p \theta_j h_j^2)$. Here $\boldsymbol{\theta}$ are the correlation parameters used in the global trend (22), $b \in [0,1]$ is an extra bandwidth parameter such that $\boldsymbol{g}_b(\boldsymbol{x}) \to \mathbf{1}$ as $b \to 0$, and $\boldsymbol{g}_b(\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{x})$ if $b = 1$. Since $\boldsymbol{g}(\boldsymbol{x})$ is the correlation of the global trend, the underlying assumption behind (25) is that whenever two points in the global trend are strongly correlated, their variances also tend to be more related. The bandwidth parameter b adds additional flexility in controlling the smoothness of the variance function: when equaling zero, it smoothes out $v(\boldsymbol{x})$ to a constant function even if the global trend is not flat.

From the $v(\boldsymbol{x})$ model in (25), we can evaluate $\hat{v}_i = v(\boldsymbol{x}_i)$ for $i = 1,\ldots,n$ and update the matrix $\boldsymbol{\Sigma} = diag\{\hat{v}_1,\ldots,\hat{v}_n\}$. Since $v(\boldsymbol{x})$ and $\boldsymbol{\Sigma}$ are the standardized local volatilities, we also need to re-scale them as

$$\boldsymbol{\Sigma} \leftarrow \boldsymbol{\Sigma}/(\frac{1}{n}\sum_{i=1}^n \hat{v}_i) \qquad \text{and} \qquad v(\boldsymbol{x}) \leftarrow v(\boldsymbol{x})/(\frac{1}{n}\sum_{i=1}^n \hat{v}_i). \tag{26}$$

This standardization makes the diagonal elements of $\boldsymbol{\Sigma}$ having unit mean, which is essential for keeping the ratio of σ^2 to τ^2 consistent in the global trend. By plugging the updated (and standardized) $\boldsymbol{\Sigma}$ back into (22), we can repeat the above process for a few more times. Usually three or four iterations are sufficient to stabilize the volatility estimates. This iterative estimation for variance is similar in spirit to the *iteratively reweighted least squares* method in classical regression.

Before concluding this section, we want to emphasize that the estimation of $v(\boldsymbol{x})$ does not need to be separately carried out before fitting the CGP model; instead, it can be seamlessly nested as an inner loop in estimating the whole model. The $v(\boldsymbol{x})$ function above is uniquely determined by the unknown parameters $\boldsymbol{\theta}$ and b. Since its correlation parameter $\boldsymbol{\theta}$ are always paired and synchronized with that of the global trend, inclusion of this volatility function $v(\boldsymbol{x})$ only adds one more parameter b to the whole model.

2.4 Estimation

In this section, we derive maximum-likelihood estimators (MLEs) for the unknown parameters in the CGP model. As suggested at the end of previous section, given each

set of $(\lambda, \mu, \tau^2, \boldsymbol{\theta}, \boldsymbol{\alpha}, b)$ values, $v(\boldsymbol{x})$ and $\boldsymbol{\Sigma} = diag\{\hat{v}_1, \ldots, \hat{v}_n\}$ values can be uniquely determined by nesting a small inner loop in the likelihood function.

Based on the multivariate normal assumptions in Section 2.3.2, the log-likelihood function (up to an additive constant) can be written as

$$
\begin{aligned}
&l(\mu, \tau^2, \sigma^2, \boldsymbol{\theta}, \boldsymbol{\alpha}, b) \\
&= -\frac{1}{2}\log(\det(\tau^2\boldsymbol{G} + \sigma^2\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})) - \frac{1}{2}(\boldsymbol{y} - \mu\mathbf{1})^\top(\tau^2\boldsymbol{G} + \sigma^2\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}(\boldsymbol{y} - \mu\mathbf{1}).
\end{aligned}
$$

Due to the invariant property of MLE under transformations, we can re-parameterize $\lambda = \sigma^2/\tau^2$ in the log-likelihood as

$$
\begin{aligned}
&l(\lambda, \mu, \tau^2, \boldsymbol{\theta}, \boldsymbol{\alpha}, b) \qquad (27)\\
&= -\frac{1}{2}[n\log(\tau^2) + \log(\det(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})) + (\boldsymbol{y} - \mu\mathbf{1})^\top(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}(\boldsymbol{y} - \mu\mathbf{1})/\tau^2].
\end{aligned}
$$

Since $\boldsymbol{\Sigma} = diag\{\hat{v}_1, \ldots, \hat{v}_n\}$ can be known through the procedures presented in the last section, the MLEs for μ and τ^2 can be easily derived from (27) as

$$\hat{\mu}(\lambda, \boldsymbol{\theta}, \boldsymbol{\alpha}, b) = (\mathbf{1}^\top(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}\mathbf{1})^{-1}(\mathbf{1}^\top(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}\boldsymbol{y}), \qquad (28)$$

$$\hat{\tau}^2(\lambda, \boldsymbol{\theta}, \boldsymbol{\alpha}, b) = \frac{1}{n}(\boldsymbol{y} - \hat{\mu}\mathbf{1})^\top(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}). \qquad (29)$$

After substituting these values into (27), we can obtain the MLEs for $(\lambda, \boldsymbol{\theta}, \boldsymbol{\alpha}, b)$ by minimizing the following (negative) log profile likelihood

$$\phi(\lambda, \boldsymbol{\theta}, \boldsymbol{\alpha}, b) = n\log(\hat{\tau}^2(\lambda, \boldsymbol{\theta}, \boldsymbol{\alpha}, b)) + \log(\det(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})), \qquad (30)$$

where $\lambda \in [0, 1], b \in [0, 1]$, $\theta_j \in [0, \alpha^l]$ and $\alpha_j \in [\alpha^l, \infty]$ for $j = 1, \ldots, p$.

For p input variables, the above likelihood function contains $2p + 2$ unknown parameters. Compared to the stationary GP model whose likelihood contains only p unknown parameters, the CGP model becomes more difficult to estimate when the input dimension p gets large. To mitigate this disadvantage, we can further assume

$$\alpha_j = \theta_j + \kappa, \ j = 1, \ldots, p, \qquad (31)$$

for the correlation parameters, which reduces the p unknown parameters $(\alpha_1, \ldots, \alpha_p)$ into a single κ. By substituting (31) into (30), the CGP only involves $p + 3$ unknown

parameters $(\lambda, \boldsymbol{\theta}, \kappa, b)$, whose MLEs can be obtained by minimizing

$$\phi(\lambda, \boldsymbol{\theta}, \kappa, b) = n \log(\hat{\tau}^2(\lambda, \boldsymbol{\theta}, \kappa, b)) + \log(\det(\boldsymbol{G} + \lambda \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2})), \tag{32}$$

subject to the constraints $\lambda \in [0,1], b \in [0,1]$, $\kappa \in [\alpha^l, \infty]$ and $\theta_j \in [0, \alpha^l]$ for $j = 1, \ldots, p$.

We now provide a general guideline for choosing the bound α^l. The idea is to specify the value of α^l based on the space-filling properties of the design points. Suppose the design $D = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ has been standardized into the unit region of $[0,1]^p$, and then define the following harmonic-type average inter-point distance d_{avg} to measure its space-filling properties (Ba and Joseph 2011)

$$d_{avg} = \Big(\frac{2}{n(n-1)} \sum_{1 \le i < k \le n} \frac{1}{d(\boldsymbol{x}_i, \boldsymbol{x}_k)^2} \Big)^{-\frac{1}{2}},$$

where $d(\boldsymbol{x}_i, \boldsymbol{x}_k) = \sqrt{(\sum_{j=1}^{p} (x_{ij} - x_{kj})^2)}$. When we assume $\theta_j = \theta$ and $\alpha_j = \alpha$ $(j = 1, \ldots, p)$ in the Gaussian correlation functions (14), correlations between points with distance d_{avg} are $g(\theta) = \exp(-\theta d_{avg}^2)$ and $l(\alpha) = \exp(-\alpha d_{avg}^2)$ for the global and local processes respectively. Because $\exp(-\alpha d_{avg}^2) \le \exp(-\alpha^l d_{avg}^2) \le \exp(-\theta d_{avg}^2)$, our recommendation for choosing α^l is to set $\exp(-\alpha^l d_{avg}^2) = 0.01$, which leads to

$$\alpha^l = \frac{\log 100}{d_{avg}^2}. \tag{33}$$

This bound is used for estimation throughout the chapter.

2.5 Properties

2.5.1 Improved Prediction for Sparse Dataset

As discussed in Section 2.1, the ordinary kriging predictor tends to revert to the global mean in regions where data are not available. This erratic phenomenon will be even more pronounced if the design points are sparse and cannot cover the input region reasonably well. The new predictor, however, relaxes the constant mean restriction in ordinary kriging and introduces another GP for modeling the mean. This global trend (mean model) is non-interpolating but smooth, which makes it immune to the erratic reversion problem in the data sparse region. Consider again the simple test function

in Figure 13, where the ordinary kriging predictor ($\hat{\theta} = 400$) appears to be erratic. When the proposed CGP model is fitted ($\hat{\lambda} = 0.07, \hat{\theta} = 143.6, \hat{\alpha} = 1892.1, \hat{b} = 1$), its global trend is shown as the dotted line in Figure 14. Although it incurs large errors around data points in region $x \in [0, 0.4]$, it behaves well in the sparse region $[0.4, 1]$ due to the smoothness property. The final CGP predictor after incorporating the local trend is shown as the dashed line in Figure 14. It can be seen that this predictor eliminates all the non-interpolating errors at design points. At locations far from data points, it tends to revert to the smooth global trend instead of a global constant, which avoids the erratic problem as in Figure 13 and yields much improved prediction. This shows the advantage of using the CGP predictor when data points are sparse in some parts of the design region. In practice, the sparseness of data points is quite common when input dimensions are high or a non-space-filling design is used.

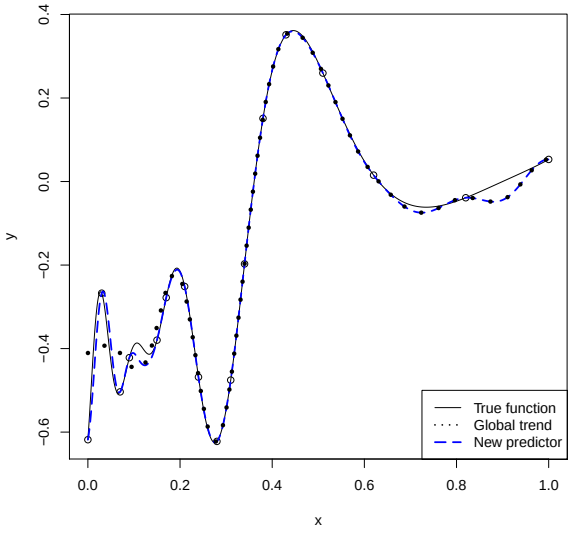

Figure 14: Plot of function $y(x) = \sin(30(x-0.9)^4)\cos(2(x-0.9)) + (x-0.9)/2$, the global trend and the CGP predictor.

2.5.2 Numerical Stability

One well-documented problem with the GP model is the potential numerical instability when computing the inverse of its $n \times n$ correlation matrix $\boldsymbol{R}$. This correlation

matrix can easily become ill-conditioned, for example, when sample size n is large, design points are close to each other, or the sample points get highly correlated while we search for the optimal correlation parameters (Ababou, Bagtzoglou and Wood 1994, Haaland and Qian 2012, Peng and Wu 2012). A near-singular correlation matrix in kriging will lead to serious numerical problems, which causes the resulting predictor unstable and unreliable.

To overcome this ill-conditioned problem, the popular approach is to add a non-zero nugget to the diagonal elements of the correlation matrix such that $\boldsymbol{R} \rightarrow (\boldsymbol{R} + \lambda \boldsymbol{I})$. Because including non-zero nugget has the inevitable drawback of making predictors over-smooth (non-interpolating), in this approach we need to reconcile the gains in numerical stability with the losses in interpolation property, and choose a trade-off value for the nugget (Ranjan, Haynes and Karsten 2012, Peng and Wu 2012).

As shown at the end of Section 2.3.1, the correlation matrix to invert in the proposed CGP model is $(\boldsymbol{G} + \lambda \boldsymbol{L})$. (Cases after including the variance matrix $\boldsymbol{\Sigma}$ remain similar.) Since the lower bounds for $\boldsymbol{\alpha}$ in (14) are moderately large and we have $\boldsymbol{L} \approx \boldsymbol{I}$, the λ in $(\boldsymbol{G} + \lambda \boldsymbol{L})$ automatically inflates the diagonal elements of the correlation matrix so that it is naturally resistent to become singular. In addition, different from the previous nugget case, the CGP model is always an interpolator and the λ value here can be freely estimated. In fact, whenever a traditional GP model has to include a non-zero nugget for numerical reasons, the CGP model can always improve it at least by removing its non-interpolating errors with a augmented $Z_{local}(\boldsymbol{x})$. This potential improvement is shown in next subsection.

2.5.3 Connection With the Nugget Predictor

To emulate deterministic outputs from computer experiments, Gramacy and Lee (2011) advocate always including a non-zero nugget in the kriging predictor for reasons even beyond computations. They argue that when model assumptions are violated or data points are sparse, the traditional GP predictor may lead to unpleasant results. Although adding a non-zero nugget to the predictor incurs extra errors around data

points, it can be crucial for fitting a well-behaved (i.e. smooth) surface and avoiding erratic predictions in the unknown region. In a variety of situations, Gramacy and Lee (2011) show that overall this non-interpolating predictor can achieve better prediction accuracy.

Interestingly, when the local process in CGP has zero correlation ($\boldsymbol{L} = \boldsymbol{I}$), its global trend just degenerates to a kriging predictor with nugget, and in this case the CGP predictor becomes $\hat{y}(\boldsymbol{x}) = \hat{y}_{nugget}(\boldsymbol{x}) + \hat{y}_{local}(\boldsymbol{x})$. In regions away from design points, since $\boldsymbol{l}(\boldsymbol{x}) = \boldsymbol{0}$ and $\hat{y}_{local}(\boldsymbol{x}) = 0$ for $\boldsymbol{x} \neq \boldsymbol{x}_i$ $(i = 1, 2, \ldots, n)$, the CGP model exactly matches the nugget predictor $\hat{y}_{nugget}(\boldsymbol{x})$. At the n design points, however, due to $\boldsymbol{l}(\boldsymbol{x}) = \boldsymbol{e}_i$ for $\boldsymbol{x} = \boldsymbol{x}_i$ $(i = 1, 2, \ldots, n)$, the $\hat{y}_{local}(\boldsymbol{x})$ still corrects the global trend and adjusts the CGP to interpolate all the data points. Just as the universal kriging generalizes the polynomial regression for interpolation, the CGP model can be similarly viewed as a generalization/improvement of the nugget predictor which eliminates errors at design points. When correlations in the local process of CGP are further estimated as positive, the above adjustments around data points tend to be continuous and smooth, which leads to a final CGP predictor inheriting the advantages from both the nugget predictor and the interpolating predictor.

Figure 15(a) demonstrates a simulated example from Gramacy and Lee (2011), where the test function $y(x) = \sin(10\pi x)/(2x) + (x-1)^4$ is evaluated at 20 unequally spaced locations to represent the sparseness of data points. Clearly, we can see that in this example the ordinary kriging predictor ($\hat{\theta} = 45.97$) makes predictions well outside the range of test function in many regions. The nugget predictor suggested by Gramacy and Lee (2011) is shown in Figure 15(b). Although non-interpolating, the nugget predictor overall gives smooth and reasonably good predictions, which reduces the *root mean squared prediction error* (RMSPE) from the previous 0.55 to 0.35. Here the RMSPE $= [\frac{1}{N}\sum_{i=1}^{N}\{\hat{y}(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)\}^2]^{1/2}$ is computed based on $N = 5000$ randomly sampled data points from the design region. Now we further consider fitting the CGP model to this example. As shown in Figure 15(c), if we assume very small correlations in $Z_{local}(\boldsymbol{x})$, the new predictor remains almost the same as the nugget predictor within most regions; when it comes to around the design points, however, the predictor jumps

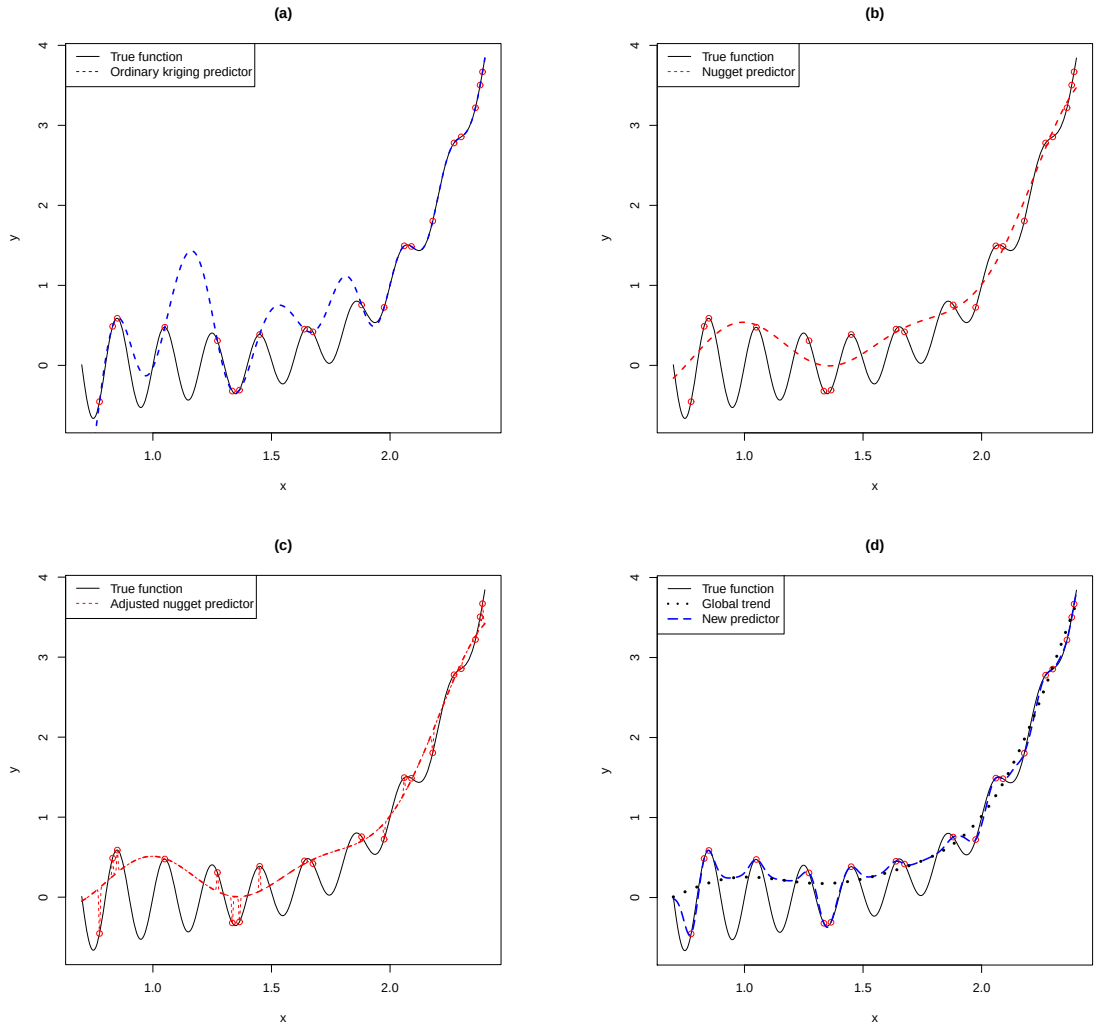

Figure 15: Plot of function $y(x) = \sin(10\pi x)/(2x) + (x-1)^4$ with (a) the ordinary kriging predictor; (b) the kriging with nugget predictor; (c) the nugget predictor with adjustments around design points; (d) the optimized CGP predictor and its global trend.

to interpolate the data, which slightly reduces the RMSPE to 0.34. After we also fully estimate the correlations in $Z_{local}(\boldsymbol{x})$ and incorporate a variance model, Figure 15(d) gives the final CGP predictor ($\hat{\lambda} = 0.019, \hat{\theta} = 2.44, \hat{\alpha} = 578.09, \hat{b} = 1$), which is smooth and gives a RMSPE as low as 0.25.

2.5.4 Improved Prediction Intervals

Apart from prediction, another frequently noted drawback of ordinary kriging is the poor coverage of its prediction intervals (Yamamoto 2000, Xiong et al. 2007, Gramacy and Lee 2011, Joseph and Kang 2011). By assuming a constant variance σ^2 throughout the whole input region, the $(1-\alpha)$ prediction interval at location $\boldsymbol{x}$ for ordinary kriging is given by

$$\hat{y}(\boldsymbol{x}) \pm z_{\alpha/2}\sigma\{1 - \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{R}^{-1}\boldsymbol{r}(\boldsymbol{x}) + \frac{(1 - \boldsymbol{r}^\top(\boldsymbol{x})\boldsymbol{R}^{-1}\mathbf{1})^2}{\mathbf{1}^\top\boldsymbol{R}^{-1}\mathbf{1}}\}^{1/2},$$

where $z_{\alpha/2}$ is the upper $\alpha/2$ critical value of the standard normal distribution. This prediction interval is often too restrictive and inadequate to cover some complex underlying surfaces since it fails to take into account the change of local variability in the design region. One typical example is demonstrated in Figure 16(a), where the test function fluctuates around zero with decreasing amplitude. The corresponding prediction intervals from ordinary kriging ($\hat{\theta} = 24.6$), however, yield the same variability pattern throughout the whole design region, which are obviously too narrow to cover the high volatility region in the left part, but also end up unnecessarily wide in the right part of the input region where the true function is almost flat. In this subsection, we introduce the prediction intervals for CGP models. By relaxing the constant variance restriction, these prediction intervals are self-adjusted according to the local variability, and can be expected to give much improved coverage.

In a Bayesian framework, the assumptions for a CGP model in (18) can be viewed as putting a prior distribution $y(\boldsymbol{x})|\mu \sim GP(\mu, \tau^2 g(\cdot) + \sigma^2(\boldsymbol{x})l(\cdot))$ on the function, which leads to the first-stage conditional distribution

$$\begin{pmatrix} y(\boldsymbol{x}) \\ \boldsymbol{y} \end{pmatrix} \Bigg| \mu \sim N_{1+n}\left[\begin{pmatrix} \mu \\ \mu\mathbf{1} \end{pmatrix}, \tau^2 \begin{pmatrix} 1 + \lambda v(\boldsymbol{x}) & \boldsymbol{q}^\top(\boldsymbol{x}) \\ \boldsymbol{q}(\boldsymbol{x}) & \boldsymbol{Q} \end{pmatrix}\right],$$

where $\lambda = \sigma^2/\tau^2$, $\boldsymbol{q}(\boldsymbol{x}) = \boldsymbol{g}(\boldsymbol{x}) + \lambda v^{1/2}(\boldsymbol{x})\boldsymbol{\Sigma}^{1/2}\boldsymbol{l}(\boldsymbol{x})$, $\boldsymbol{Q} = \boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2}$ and all the other notations remain the same as in Section 2.3.2. Here for simplicity, the variance and correlation parameters are assumed to be known. If we further assume a second-stage noninformative prior for μ: $p(\mu) \sim 1$ and integrate it out, then the predictive distribution for $y(\boldsymbol{x})$ can be derived as

$$y(\boldsymbol{x})|\boldsymbol{y} \sim N_1(\mu_{0|n}(\boldsymbol{x}), v^2_{0|n}(\boldsymbol{x}))$$

where

$$\mu_{0|n}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{q}^\top(\boldsymbol{x})\boldsymbol{Q}^{-1}(\boldsymbol{y} - \hat{\mu}\boldsymbol{1}) \qquad \text{for} \qquad \hat{\mu} = (\boldsymbol{1}^\top\boldsymbol{Q}^{-1}\boldsymbol{1})^{-1}(\boldsymbol{1}^\top\boldsymbol{Q}^{-1}\boldsymbol{y}),$$

and

$$v^2_{0|n}(\boldsymbol{x}) = \tau^2\{1 + \lambda v(\boldsymbol{x}) - \boldsymbol{q}^\top(\boldsymbol{x})\boldsymbol{Q}^{-1}\boldsymbol{q}(\boldsymbol{x}) + \frac{(1 - \boldsymbol{q}^\top(\boldsymbol{x})\boldsymbol{Q}^{-1}\boldsymbol{1})^2}{\boldsymbol{1}^\top\boldsymbol{Q}^{-1}\boldsymbol{1}}\}. \tag{34}$$

The derivation for these results is tedious but standard, which follows similar development steps as in Santer et al. (2003, Chapter 4.3). It can be seen that our previously proposed predictor in (20) is nothing but the posterior mean of the function given the data. Now a (pointwise) prediction interval for this predictor can be constructed by

$$\hat{y}(\boldsymbol{x}) \pm z_{\alpha/2} v_{0|n}(\boldsymbol{x}), \tag{35}$$

where $z_{\alpha/2}$ is the upper $\alpha/2$ critical value of the standard normal distribution.

Note that, since $\boldsymbol{q}^\top(\boldsymbol{x}_i)\boldsymbol{Q}^{-1} = \boldsymbol{e}_i^\top$ and $\boldsymbol{e}_i^\top\boldsymbol{q}(\boldsymbol{x}_i) = 1 + \lambda v(\boldsymbol{x}_i)$, the above posterior variance $v^2_{0|n}(\boldsymbol{x})$ equals zero whenever $\boldsymbol{x} = \boldsymbol{x}_i$ for $i = 1, \ldots, n$. Thus, as in ordinary kriging, the width of the prediction interval shrinks to zero at each data point, which is quite intuitive since both models interpolate the responses at each observed location. On the other hand, however, different from ordinary kriging, the variance of predictive distribution in (34) depends on the local variability of the underlying surface, which intrinsically adjusts the widths of the prediction interval. Consider again the test function in Figure 16. It can be seen in Figure 16(b) that the prediction intervals from a CGP model ($\hat{\theta} = 2.1, \hat{\alpha} = 54.85, \hat{\lambda} = 1, \hat{b} = 1$) become much wider in the left region when the function fluctuates rapidly, but quickly narrow down as the underlying function becomes flat. Compared with the prediction intervals for ordinary kriging, the

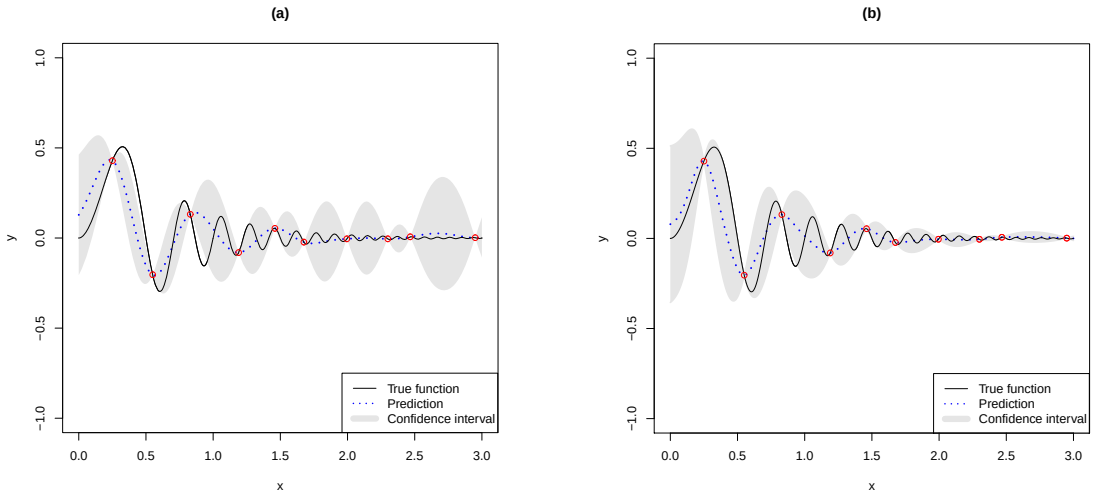

Figure 16: Plot of function $y(x) = \exp(-2x)\sin(4\pi x^2)$ and the prediction intervals from (a) ordinary kriging; (b) the CGP model.

new intervals can more precisely demonstrate the change of prediction uncertainties throughout the input region: i.e. the predictive variances are much larger in the left part of region than in the right. One way to quantify such improvements is through computing the *interval score* for central prediction intervals (Gneiting and Raftery 2007) which is defined as $S_\alpha^{int}(l,u;x) = (u-l) + \frac{2}{\alpha}(l-x)\mathbb{1}\{x<l\} + \frac{2}{\alpha}(x-u)\mathbb{1}\{x>u\}$ for a $(1-\alpha)\%$ central prediction interval $[l,u]$. This scoring rule (to be minimized) rewards narrow prediction intervals and also penalizes lack of coverage. For the prediction intervals in Figure 16, the average interval score (based on 3000 randomly sampled test points) for the ordinary kriging in (a) is 0.62 while for the CGP model in (b) is only 0.32, which shows almost 50% improvements.

2.5.5 Extensions to Noisy Data

In the previous sections, we model the deterministic outputs from a computer experiment by coupling two GPs. As an extension to this, sometimes it is also possible to use the sum of more than two GPs for gaining additional flexibility in the model and satisfying special needs. One important application of this extension is to modify the new predictor for modeling data with random errors.

Based on the previous model form in Section 2.3.2, we can add a third GP (with

zero correlation) to account for the white noise as follows

$$Y(\boldsymbol{x}) = Z_{global}(\boldsymbol{x}) + \sigma(\boldsymbol{x}) Z_{local}(\boldsymbol{x}) + \varepsilon(\boldsymbol{x}),$$

where $Z_{global}(\boldsymbol{x})$, $Z_{local}(\boldsymbol{x})$ are the same stationary GPs as in (18), and the error term $\varepsilon(\boldsymbol{x})$ is assumed to be $N(0, \sigma_\varepsilon^2(\boldsymbol{x}))$ distributed, uncorrelated at different input locations and also independent of the other two GPs. Suppose the error variances $\Sigma_\varepsilon = diag\{\sigma_\varepsilon^2(\boldsymbol{x}_1), \ldots, \sigma_\varepsilon^2(\boldsymbol{x}_n)\}$ are given, then the best linear unbiased predictor can be easily updated by modifying (20) as follows

$$\begin{aligned}\hat{y}(\boldsymbol{x}) =& \hat{\mu} + (\tau^2 \boldsymbol{g}(\boldsymbol{x}) + \sigma^2 v^{1/2}(\boldsymbol{x}) \boldsymbol{\Sigma}^{1/2} \boldsymbol{l}(\boldsymbol{x}))^\top (\tau^2 \boldsymbol{G} + \sigma^2 \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2} + \Sigma_\varepsilon)^{-1} (\boldsymbol{y} - \hat{\mu}\mathbf{1}) \\ =& \hat{\mu} + (\boldsymbol{g}(\boldsymbol{x}) + \lambda v^{1/2}(\boldsymbol{x}) \boldsymbol{\Sigma}^{1/2} \boldsymbol{l}(\boldsymbol{x}))^\top (\boldsymbol{G} + \lambda \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2} + \rho \Sigma_\varepsilon)^{-1} (\boldsymbol{y} - \hat{\mu}\mathbf{1}),\end{aligned}$$

where $\rho = 1/\tau^2$, $\hat{\mu} = (\mathbf{1}^\top (\boldsymbol{G} + \lambda \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2} + \rho \Sigma_\varepsilon)^{-1} \mathbf{1})^{-1} (\mathbf{1}^\top (\boldsymbol{G} + \lambda \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2} + \rho \Sigma_\varepsilon)^{-1} \boldsymbol{y})$ and all the other notations remain the same as in (20). This predictor for noisy data is no longer an interpolator, and its parameter estimation can be similarly carried out as in the previous sections, except for $(\boldsymbol{G} + \lambda \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2})$ replaced by $(\boldsymbol{G} + \lambda \boldsymbol{\Sigma}^{1/2} \boldsymbol{L} \boldsymbol{\Sigma}^{1/2} + \rho \Sigma_\varepsilon)$ in the models.

2.6 Examples

Example 1. For any non-stationary modeling approach, one commonly raised concern is that if the true surface is indeed a realization from a stationary Gaussian process, whether the "unnecessarily sophisticated" non-stationary modeling approach can perform as good as the "correct" stationary model. To test the performance of our proposed model in such cases, we simulate sample paths from various two-dimensional stationary Gaussian processes for 50 times, and fit both the CGP and the stationary GP models to each of them for comparison. A 24-run maximin distance Latin Hypercube Design (LHD) is used in these simulations, and for each time the true correlation parameters in GP are randomly generated from $[1, 5]$. In each iteration, once the design and correlation parameters are fixed, a 24×24 correlation matrix $\boldsymbol{R}$ is uniquely determined. A sample path from the corresponding stationary GP can then be drawn by simulating a random sample vector from the multivariate normal distribution $N_n(\mu \mathbf{1}^n, \sigma^2 \boldsymbol{R}^n)$ with $n = 24, \mu = 0, \sigma^2 = 1$.

Table 1: RMSPE values for three predictors based on 5000 testing data.

Method	Maximin LHD	Adaptive Design
GP	0.188	0.266
CGP	0.144	0.159
TGP	0.312	0.465

After drawing stationary sample paths as above for 50 times, we fit CGP models to each of them. Among the 50 fitted models, 42 out of them have $\hat{\lambda} = 0$, which shows that the CGP has perfectly degenerated to the stationary GP model. For the other eight CGP models, their $\hat{\lambda}$ values are also extremely small, with the largest one only as 0.003. Measured by the leave-one-out cross validation error, the prediction accuracy of CGP model and the stationary GP model are almost identical in these cases.

Example 2. In this example, we provide two test functions possessing non-stationary features: one in two dimensions and the other has 10-dimensional inputs. The first function is the two-dimensional $f(x_1, x_2) = \sin(1/(x_1 x_2))$, $(x_1, x_2 \in [0.3, 1])$, whose surface fluctuates rapidly when x_1 or x_2 is small, but gradually becomes smooth as x_1 and x_2 increase toward one. The second test function (known as the Michalewicz's function) is in 10 dimensions, which has the following form:

$$f(\boldsymbol{x}) = -\sum_{i=1}^{10} \sin(x_i)[\sin(\frac{ix_i^2}{\pi})]^{2m}, 0 \leq x_i \leq \pi, i = 1, \ldots, 10.$$

Typically, this function is used with $m = 10$, which leads to a high dimensional surface containing many local optima, and its volatility varies dramatically throughout the input region.

We use a 24-run maximin distance LHD and a 24-run adaptive design from Xiong et. al (2007) to evaluate the first test function. Both the GP and CGP models are fitted to these two designs, and their RMSPEs are compared based on additional 5000 randomly sampled testing data. From the results in Table 1, we can see that the CGP predictor improves the accuracy of the GP predictor by 23% and 40% for each design. In Table 1, we also fit the Bayesian treed Gaussian process (TGP) model (Gramacy

and Lee 2008) to the two designs for comparison. The RMSPEs of this non-stationary treed model are relatively large, which probably are due to its inefficient partitioning of the input region.

To further test the performance of CGP predictor based on different designs, we generate fifty 100-run random LHDs to evaluate the second test function, and fit the GP and CGP models to each of them. RMSPEs of the two predictors are plotted in Figure 17 for the 50 random designs. It can be seen that, compared to the GP model, the CGP predictor can always give better approximations to this complex surface based on any random LHD. The RMSPEs of the two predictors based on a 100-run maximin distance LHD are also marked in this plot.

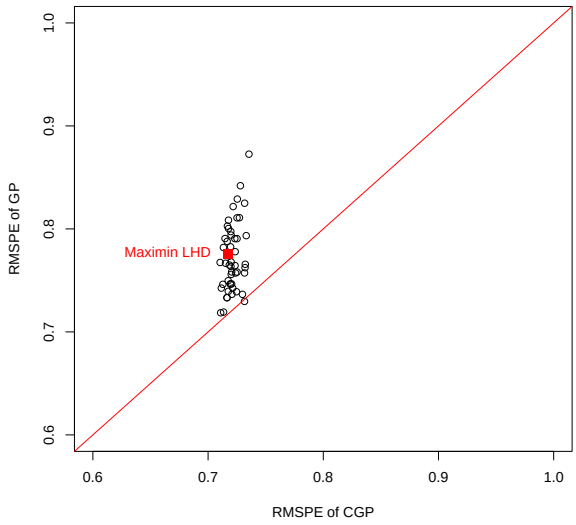

Figure 17: RMSPEs of GP and CGP models for the Michalewicz's function. Points falling above the diagonal line indicating larger prediction errors for the GP model.

Example 3. Qian et al. (2006) described a computer simulation of a heat exchanger for electronic cooling applications. The device under study consists of linear cellular materials and is used for dissipating the heat generated by some sources such as a microprocessor. The response of interest is the total rate of steady state heat transfer of the device, which depends on the mass flow rate of entry air

$\dot{m} \in (0.00055, 0.001)$, the temperature of entry air $T_{in} \in (270, 303.15)$, the solid material thermal conductivity $k \in (330, 400)$ and the temperature of the heat source $T_{wall} \in (202.4, 360)$. The device is assumed to have fixed overall width (W), depth (D), and height (H) of 9, 25, and 17.4 millimeters, respectively. In Qian et al. (2006), the study involved two types of simulators: an expensive finite element simulator and a relatively cheaper finite difference simulator. Since the latter type of simulation was systematically conducted in the design space while the previous one only available at limited locations, here we only focus on using the finite difference simulation results to compare the prediction accuracy of several different models. Because the four input variables are in very different scales, all of them are standardized into the (0,1) region before analysis.

Qian et al. (2006) used a 64-run orthogonal array-based Latin Hypercube design for running the finite difference simulations with an extra 14-run test data set for assessing the predictions from surrogate model. If no prior information is available for the function and an ordinary kriging with Gaussian correlation function is directly fitted, the maximum likelihood estimates for its correlation parameters are (0.22, 4.37, 0.14, 7.24), which yield a RMSPE of 5.15. However, for this particular problem, the physical domain knowledge indicates that a linear component is very likely to exist between the response and factors. As a result, Qian et al. (2006) included the linear trend into the model and fitted a universal kriging to the data. Their results showed that the linear effects for T_{in} and T_{wall} are significant but for the other two variables are almost negligible. By including these two linear effects into the global trend, the RMSPE can be successfully reduced to only 2.588. Now we fit a CGP model to the data for comparison. Based on the maximum likelihood method in Section 2.4, we can estimate the unknown parameters as $\hat{\boldsymbol{\theta}}$=(0.008,0.3,0.01,11.74), $\hat{\boldsymbol{\alpha}}$=(11.81,12.17,11.94,23.48), $\hat{\lambda}$=0.019 and $\hat{b}$=1. The RMSPE for this new predictor is 2.24, which is much better than the ordinary kriging and even smaller than the previous improved result from universal kriging. Note that in the global trend of this new predictor, the two correlation parameters $\hat{\theta}_2$ and $\hat{\theta}_4$ (for T_{in} and T_{wall}) are remarkably larger than the others, which perfectly coincides with the two significant

linear trends in universal kriging. This demonstrates the effectiveness of CGP model for capturing the global trend. In most common situations where no functional relationship in the global trend can be known in advance, the ability to automatically estimate the trend and the variance is a great advantage for the new predictor over the other methods.

2.7 Conclusions

In this chapter, we present an intuitive approach for approximating complex surfaces that are not second-order stationary. The new predictor intrinsically incorporates a global trend and a flexible variance model, and all of its parameters can be estimated in a single stage. Compared with many existing methods, the new model enjoys several advantages such as numerical stability, improved prediction accuracy and flexible prediction intervals. R codes for fitting the CGP model can be obtained from the authors' website.

For modeling the non-stationarity in variance, one reviewer draws our attention to a related idea called *scaling* in the geostatistical literature (Banerjee, Charlin, and Gelfand 2003). The scaling approach is given in the form $Y(\boldsymbol{x}) = \sigma(\boldsymbol{x})Z(\boldsymbol{x})$, where $Z(\boldsymbol{x})$ denotes a stationary process and $\sigma^2(\boldsymbol{x})$ is a variance function that needs to be specified. By choosing $\sigma^2(\boldsymbol{x})$ as the exponent of another Gaussian process, Huang, Wang, Breidt and Davis (2011) proposed a *stochastic heteroscedastic process* (SHP) model $y(\boldsymbol{x}) = \boldsymbol{g}^\top(\boldsymbol{x})\boldsymbol{\beta} + \sigma \exp(\tau\alpha(\boldsymbol{x})/2)Z(\boldsymbol{x})$ for low-dimensional environmental applications, where $\alpha(\boldsymbol{x})$ is defined to be another stationary Gaussian process that is independent of $Z(\boldsymbol{x})$. Although this SHP model does not have a flexible global trend, its variance model is more sophisticated than our CGP model. This additional flexibility in variance, however, comes with the expenses of a very difficult and complicated estimation procedure. Since the likelihood function of the SHP model has no closed-form expression, simulation-based approximations have to be applied for the likelihood value during each step of its optimization. Obviously, this can be computationally very challenging (or even infeasible) when the dimension of unknown parameters is high, which limits its application in computer experiments.

Recently, we also noticed an interesting work from Haaland and Qian (2011), which uses the sum of multiple GPs to emulate outputs from large scale computer experiments. However, the purposes of their work is different from ours. The aim of Haaland and Qian (2011) is mainly to control the numerical error in computing interpolators based on huge amount of data. Their multiple GP models are fitted sequentially and each of them is only based on a subset of data points. On the contrary, our method is developed to improve the precision in modeling expensive simulation results that are not second-order stationary. Both our global and local GPs are fitted based on the entire data set and all parameters in our model are also estimated in a single stage.

For p input factors, the proposed CGP model involves $p+3$ unknown parameters, which is computationally slightly more expensive to fit than the ordinary kriging. This is the price we need to pay for incorporating the extra flexility in modeling the global trend and the change of variance. We want to note that although the number of parameters in ordinary kriging can also be extended from p to $2p$ by generalizing its Gaussian correlation function to the *power exponential correlation function* $r(\boldsymbol{h}|\boldsymbol{\theta}, \boldsymbol{w}) = \exp(-\sum_{j=1}^{p} \theta_j |h_j|^{w_j})$ or even a *Matern correlation function*, this extension alone cannot solve the problems discussed in this chapter, since the resulting predictor still remains second-order stationary.

2.8 Appendix: Proof of Theorem 4

Since both the single-stage predictor (21) and the sequential predictor (24) contain the same global trend $\hat{y}_{global}(\boldsymbol{x})$ as in (22), we only need to prove $\hat{y}_{local}(\boldsymbol{x}) =$

$v^{1/2}(\boldsymbol{x})\hat{y}_{adj}(\boldsymbol{x})$.

$$
\begin{aligned}
&v^{1/2}(\boldsymbol{x})\hat{y}_{adj}(\boldsymbol{x}) = v^{1/2}(\boldsymbol{x})\boldsymbol{l}^\top(\boldsymbol{x})\boldsymbol{L}^{-1}\boldsymbol{s}^* \\
&= v^{1/2}(\boldsymbol{x})\boldsymbol{l}^\top(\boldsymbol{x})\boldsymbol{L}^{-1}\boldsymbol{\Sigma}^{-1/2}[\boldsymbol{y} - \hat{\mu}\mathbf{1} - \boldsymbol{G}(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1})] \\
&= v^{1/2}(\boldsymbol{x})\boldsymbol{l}^\top(\boldsymbol{x})\boldsymbol{L}^{-1}\boldsymbol{\Sigma}^{-1/2}[\boldsymbol{I} - \boldsymbol{G}(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}](\boldsymbol{y} - \hat{\mu}\mathbf{1}) \\
&= \lambda v^{1/2}(\boldsymbol{x})\boldsymbol{l}^\top(\boldsymbol{x})\boldsymbol{\Sigma}^{1/2}(\lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}[\boldsymbol{I} - \boldsymbol{G}(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}](\boldsymbol{y} - \hat{\mu}\mathbf{1}) \\
&=^{(*)} \lambda v^{1/2}(\boldsymbol{x})\boldsymbol{l}^\top(\boldsymbol{x})\boldsymbol{\Sigma}^{1/2}(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}(\boldsymbol{y} - \hat{\mu}\mathbf{1}) \\
&= \hat{y}_{local}(\boldsymbol{x}),
\end{aligned}
$$

where the equality $=^{(*)}$ holds because $(\lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}[\boldsymbol{I} - \boldsymbol{G}(\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2})^{-1}](\boldsymbol{G} + \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{L}\boldsymbol{\Sigma}^{1/2}) = \boldsymbol{I}$.

CHAPTER III

INTEGRATING ANALYTICAL MODELS WITH FINITE ELEMENT MODELS: AN APPLICATION IN MICROMACHINING

3.1 Introduction

The prediction of cutting forces is very important in designing a mechanical micromachining process because the machine and tool deflect under the action of cutting forces, which affects the geometrical accuracy of the machined feature. In addition, the tool radii are extremely small (of the order of tens of microns) and therefore have low area moment of inertia which induces very high flexural stresses even at small cutting forces. These high stresses could potentially lead to catastrophic tool failure. Consequently, it is very important to predict the (steady state) machining forces accurately in designing the micromachining process to avoid tool breakage and ensure the geometrical accuracy of the machined feature.

In such problems, finite element simulations are frequently used for predicting machining forces and studying the effects of various process variables. By solving a set of partial differential equations numerically, the finite element models take into account the frictional contact models at tool-chip interface during cutting and also incorporate thermomechanical coupling effects accounting for the change in material response due to the increased temperature from plastic deformation. Although it is very accurate, simulating machining forces at the micro-scale level is computationally very expensive and each run can take several hours or days to complete. Therefore, it is important to carefully design the finite element simulations so that maximum information about the system can be gathered with the limited computational resource available.

Besides the computationally intensive simulations, analytical models having closed-form solutions for the output in terms of the inputs are also available in many cases for capturing the micromachining process mechanics. These analytical models are derived based on various engineering assumptions and approximations (such as the slip-line field theory) which cannot take into account the effect of temperature rise due to the heat generated from plastic deformation. As a result, the analytical models are generally less accurate than the finite element models which are based on the governing partial differential equations that capture the physics of the process in a better manner. On the other hand, however, the sacrifice of accuracy also reduces the computations, and the analytical models are capable of instantly evaluating the outputs throughout the input region with little computational cost.

Although designing for the finite element simulations alone is a well-studied topic in the computer experiments' literature (Santner et al. 2003 and Fang et al. 2006), efficient strategies when the analytical models are also present still remain unclear. In this chapter, we consider how to utilize the additional information from the analytical models to more efficiently design the finite element simulations in the micromachining process. After running those simulations, a (statistical) metamodel can be fitted to integrate the analytical models with the finite element simulation data, which can then provide accurate and instant predictions for the machining forces. In the rest of this section we first give a brief review for the two different types of models used in our micromachining process. Section 3.2 examines several traditional methods for designing the finite elements simulations and in Section 3.3, we propose a new sequential strategy which first elicits information from the analytical models and then construct designs for the finite element simulations in a more efficient manner. A detailed analysis for building the machining force metamodels is presented in Section 3.4.

3.1.1 Analytical Models

There are several ways to model the cutting forces obtained in machining. The oldest model was given by Merchant (1944) by balancing the forces on the chip, the workpiece and the tool. Additionally, mechanistic models have been used to

predict the cutting forces at the macroscale, but these models need to be calibrated experimentally which limits their utility. Modeling of microscale cutting process is challenging as the tool can no longer be assumed to be sharp and the edge radius effect has to be accounted for. The use of a slip-line field model is very useful in capturing the ploughing phenomenon which is typical of micromachining and other microscale processes such as polishing as given by Waldorf et al. (1998) and Challen and Oxley (1984). The analytical cutting force model presented here is comprised of a slip-line field based geometric model of plastic deformation arising from chip formation, a material model of the material flow strength given by Yan et al. (2007), and a force model derived from the slip-line field model given by Manjunathiah and Endres (2000). The final outputs of the analytical models are the cutting and thrust forces.

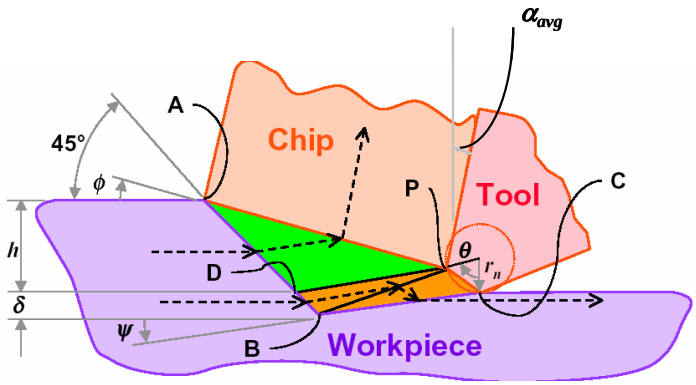

Figure 18: Geometric model of the cutting process with an edge radius tool (Manjunathiah and Endres 2000).

Figure 18 shows the slip-line field model of orthogonal cutting which describes the geometry of the plastic deformation field produced in the micromachining operation. The plastic strain ε and strain rate $\dot{\varepsilon}$ can be computed by the following equations as

given in Manjunathiah and Endres (2000) and Singh and Melkote (2009):

$$\gamma_{chip} = \frac{\sqrt{2}\sin\theta_{PD}}{\sin(\pi/4+\theta_{PD})} + \frac{\cos(\alpha_{avg}+\theta_{PD})}{\cos(\alpha_{avg}-\phi)\sin(\phi+\theta_{PD})}$$

$$\gamma_{work} = \frac{\sqrt{2}\sin\theta_{PD}}{\sin(\pi/4+\theta_{PD})} + \frac{\sin(\theta_{PD}+\theta/2)}{\sin(\theta_{PB}+\theta/2)\sin(\theta_{PB}+\theta_{PD})} + \frac{\sin\theta/2}{\sin\Psi\sin(\Psi+\theta/2)}$$

$$\dot{\gamma}_{chip} = 2V\frac{\gamma_{chip}}{\sqrt{2}\sin(\pi/4+\theta_{PD}\overline{PD})}$$

$$\dot{\gamma}_{work} = 2V\frac{\gamma_{work}}{\sqrt{2}\sin(\pi/4+\theta_{PD})\overline{PD} + \sin(\Psi+\theta/2)\overline{PC}/\sin\Psi}$$

$$\varepsilon = \frac{\gamma_{eff}}{\sqrt{3}} = \frac{\nu_{chip}\gamma_{chip}+\nu_{work}\gamma_{work}}{(\nu_{chip}+\nu_{work})\sqrt{3}}$$

$$\dot{\varepsilon} = \frac{\dot{\gamma_{eff}}}{\sqrt{3}} = \frac{\nu_{chip}\dot{\gamma_{chip}}+\nu_{work}\dot{\gamma_{work}}}{(\nu_{chip}+\nu_{work})\sqrt{3}}$$

where α_{avg} is the rake angle, V is the cutting velocity, γ is the shear strain, $\dot{\gamma}$ is the shear strain rate, γ_{eff} is the effective shear strain, $\dot{\gamma}_{eff}$ is the effective shear strain rate, ν_{chip} is the volume of chip (traingle ADP), ν_{work} is the volume of workpiece deformation (quadrilateral BDPC), θ is the chip separation angle, ϕ is the shear angle, and θ_{PD}, θ_{PB}, Ψ are the acute angles made by the lines PD, PB, BC with the horizontal line in Figure 18, respectively.

The plastic strain ε and strain rate $\dot{\varepsilon}$ from the above geometric model can then be used in a constitutive material model to evaluate the flow stress σ_f. The Johnson-Cook type multiplicative material flow stress model for H-13 steel is proposed by Yan et al. (2007) as

$$\sigma_f = (a + b\varepsilon^n + c\log(\varepsilon+\varepsilon_0) + d)(1 + E\log(\frac{\dot{\varepsilon}}{\dot{\varepsilon_0}}))(1 - (\frac{T-T_r}{T_m-T_r})^m),$$

where ε_0 is a reference strain which is taken to be 10^{-3}, $\dot{\varepsilon_0}$ is a reference strain rate typically taken to be 1 s^{-1}, T_m is the melting temperature of the material, T is the temperature in the workpiece, T_r is a reference room temperature taken to be 25^oC. Since the analytical models do not capture the temperature rise from the heat generated due to plastic deformation and tool-chip friction during machining, the T here is set to be the same as the room temperature T_r. The values of the material constants a, b, c, d, E, m and n are given in Singh and Melkote (2009).

Finally, the force model can be derived from the equilibrium of forces (cutting and thrust forces) acting on the slip-lines. Accounting for the variation in the shear flow stress S in the material removal plane, the total cutting and thrust forces F_c and F_t can be approximated as in Manjunathiah and Endres (2000)

$$F_c = \{(h-p)\cot\phi + h + r_n\sin\theta - (k-1)\delta\}S,$$
$$F_t = \{(h-p)\cot\phi - h + r_n\sin\theta - (k-1)\delta\cot\Psi\}S,$$

where $S = \sigma_f/\sqrt{3}$, h is the depth of cut, r_n is the tool edge radius, k is the normal stress factor, p is the height of the chip separation point P measured from point C in Figure 18 and δ is the depth of plastic deformation below the tool. Figure 19 shows the flow chart of predicting machining forces using the above analytical models.

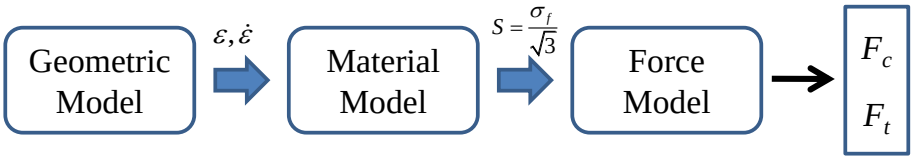

Figure 19: Flow chart of the force prediction using analytical models.

3.1.2 Finite Element Models

Commercially available finite element simulation software (DEFORM®) which contains extensively developed microstructure models has also been used to study the effect of input parameters during cutting. It uses a Lagrangian formulation and has a powerful capability of automatic remeshing which helps in creating a new mesh whenever the mesh gets distorted during the large plastic deformation process.

The tool is given a velocity in the horizontal direction to simulate orthogonal cutting. To model the heat transfer from the tool to the environment, heat exchange option is defined in DEFORM® for all the surfaces of the tool. The vertical motion is constrained at the base and the horizontal motion is constrained at the free end away from the tool. The dimension of the work piece is 5 mm × 1.5 mm with the highly dense mesh generated in the cutting region. Tungsten Carbide (WC) is used as the tool material. Figure 20 shows the mesh of finite element models and

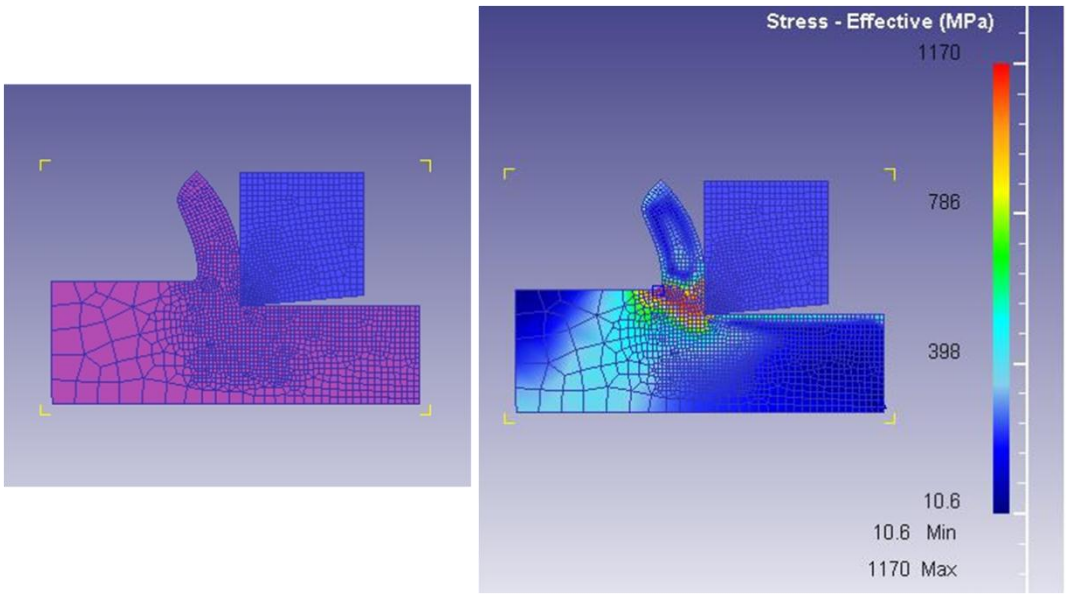

Figure 20: Finite element model for micromachining: (a) Mesh; (b) Stresses developed during machining.

the stress contours for micromachining. The modeling approach accounts for the effects of rake angle (α_{avg}), tool speed (V) and depth of cut (h) on the cutting and thrust forces. The values of the input parameters can be changed and the cutting force response can be obtained for different cutting conditions from the finite element models. Different from the typical deterministic simulations which are extensively studied in computer experiments' literature, output from the DEFORM® software is subject to a few numerical fluctuations. As a result, in our analysis the design and modeling strategies for the finite element simulations also need to take this stochastic error into account.

3.2 Existing Design Methods

Since the finite element models are time-consuming to run, our goal is to carefully select a set of input values (design) to run the finite element simulations and then fit a metamodel (surrogate model) to the simulation data to approximate the overall response surface. Traditional way to design the finite element simulations is to use *space-filling designs* (Santner et al. 2003 and Fang et al. 2006), which assume

no prior knowledge about the underlying system and tend to spread points evenly throughout the input region without replications. Typical examples include the *maximin/minimax* distance designs (Johnson, Moore and Ylvisaker 1990) and the *Latin hypercube designs* (LHDs) (McKay, Beckman and Conover 1979).

For computer experiments with two levels of accuracy, Qian et al. (2009b) proposed the concept of nested space-filling design, where a smaller design for high-accuracy experiments (such as using finite element models) are nested within a larger design for low-accuracy experiments (such as using analytical models), and both designs themselves are space-filling in low dimensions. Since a low-accuracy experiment is usually much cheaper, this structure allows for running a larger number of low-accuracy experiments to fit a base surrogate model. Furthermore, the precision of this base model can be adjusted at the nested design points where the high-accuracy simulated data are also available. Additional references on the construction of a nested space-filling design can be found in Qian (2009) and Qian et al. (2009a).

The rationale behind all the above space-filling designs is based on the belief that the important features of the response surface are as likely to be in one part of the input region as another. However, this should not be the case when the analytical models are available, because the cheap analytical models can be used to elicit critical knowledge about the underlying surface prior to conducting the expensive simulation studies. Since the traditional space-filling designs cannot be easily adjusted according to the prior knowledge, many of their runs can be in unwanted regions. This limitation motivates us to develop customized designs as in next section.

3.3 A New Design Strategy

In this chapter, we propose to perform a sensitivity analysis using the analytical models as the first step and then utilize these preliminary results to more efficiently design the finite element simulations. Many standard procedures for conducting the sensitivity analysis are well-established in the literature and have been documented in books such as Saltelli et al. (2000) and Santner et al. (2003). One approach to perform the sensitivity analysis is to fit regression models to the outputs of analytical

model and assess the importance of each variable by their regression coefficients. The main purpose here is not to obtain a perfect regression model, but instead to identify the relative importance of each input factor and assess their factor effects. In some cases if the analytical models themselves are already in very simple forms (such as polynomials), their factor effects can even be interpreted directly without building any regression model. Note that since in this step the sensitivity analysis is only based on the low-accuracy analytical models, we need to protect for possible missing effects in later steps.

After performing the sensitivity analysis, the design for finite element simulations should be adjusted according to the elicited information for the underlying surface. One limitation for the existing space-filling designs is that they always treat input factors equally and cannot be customized to comply with the different effects of each variable. In addition, the traditional designs (such as the popular LHDs) emphasize to have nonredundant design points when they are projected onto a lower-dimensional input space to guard against possible inert factors. After performing the sensitivity analysis, however, this projection property is no longer a major concern, since we can identify possible inactive factors and assign them very low number of levels in advance. Moreover, as pointed out by some authors (such as Bingham et al. 2009, Ba and Joseph 2011), although designs with many levels are ideal to capture the complex nonlinear effects in the simulated surface, it is not essential that the number of levels for each factor must be as large as the number of runs. For example, if the sensitivity analysis suggests that the high-order nonlinear effects for some input factors tend to be insignificant, using fewer levels for them usually can increase our ability to estimate the high-order interactions. For stochastic simulations or finite element simulations subject to numerical errors, it is also more desirable to moderately reduce the number of levels. In this aspect, the marginal constraint for LHD is so restrictive that sometimes it impairs the other desirable properties of the design.

In this chapter, we propose a two-stage design which can efficiently adapt themselves to the prior information from the sensitivity analysis and assign a customized

number of levels for each input factor in the finite element simulations. The building blocks we choose for this new design are the *orthogonal arrays* (OAs) (Hedayat, Sloane and Stufken 1999), whose previous applications in designing finite element simulations are very limited mainly due to its redundancy of the design points when projected onto a subspace. In the literature, a few studies reported the connections between the OAs and the space-filling designs (Fang and Mukerjee 2000, Kerr 2001). Ba and Joseph (2011) recently proposed to split the OA into multiple layers to eliminate the projection redundancy and achieve desirable space-filling properties. In this chapter, we adopt a reverse strategy. Instead of splitting the OA into layers, we propose to construct a two-stage design which combines two different arrays.

In the first stage, we customize the number of levels for each input factor based on the results from sensitivity analysis, and a compatible OA of desirable size can be generated according to the tables in Hedayat, Sloane and Stufken (1999) and Wu and Hamada (2009) to accommodate these *basic levels* for the input variables. Due to the preliminary sensitivity analysis, projection property of this customized design is no longer a concern and the selected OA can be used as a Stage-I design which forms a good basis for designing the finite element simulations. However, one limitation of this design is that it is constructed purely based on the prior information elicited from the analytical models, which are derived based on certain assumptions and approximations. Since the finite element models can capture much more delicate features of the underlying process, their input effects may have been underestimated in the preliminary sensitivity analysis. In light of this, a Stage-II design is further needed to proportionally increase the number of levels for each input factor.

To construct the Stage-II design, we first need to decide the number of extra levels to add for each input variable. This choice can be quite flexible and mainly depends on how accurately the analytical models can resemble the finite element models. If their difference is not very substantial such as in the case of our micromachining process, an intuitive strategy is to fill in the *additional levels* at the *midpoints* between the existing basic levels for each factor. (If the difference between these two types of models are indeed substantial, practitioners may need to go back to scrutinize

the underlying assumptions of the analytical models and try to make improvements before they can proceed in this way.) For any possible inactive factor identified from the sensitivity analysis, we recommend also assigning at least one more additional level to its existing nominal level. This approach increases the number of levels for each variable proportionally (nearly doubled) from the Stage-I design and make them large enough to capture the possible complex nonlinear effects existed in the finite element simulations. Since a single OA accommodating all those basic and additional levels tend to get too large, we propose to construct the Stage-II design using a second OA which only contains the newly added levels for each input factor. When superimposing this Stage-II design onto the original Stage-I array, we can obtain a combined design in economical run size but with augmented number of levels for each input. To achieve more flexible run size, the Stage-II design can alternatively be selected as a subset of the second OA by minimizing the following space-filling measure

$$\left(\sum_{i,j \in \mathrm{II}, i \neq j} \frac{1}{d^m(\boldsymbol{x}_i, \boldsymbol{x}_j)} + \sum_{i \in \mathrm{II}, k \in \mathrm{I}} \frac{1}{d^m(\boldsymbol{x}_i, \boldsymbol{x}_k)} \right)^{1/m} \tag{36}$$

where $\boldsymbol{x}_k$ ($k \in \mathrm{I}$) represent the fixed points from the Stage-I experiments, $d^2(\cdot, \cdot)$ is the Euclidean distance between any two points, and $\boldsymbol{x}_i$ ($i \in \mathrm{II}$) correspond to the points for Stage-II design which we need to choose from the second OA. The first part in (36) maximizes the distances among the Stage-II design points and the second part in (36) maximizes the distances from the Stage-II to Stage-I design points. When the constant m is chosen to be sufficiently large, it can also be seen that the objective function in (36) becomes equivalent to the maximin distance measure (Morris and Mitchell 1995).

The two-stage design generated as above has several desirable properties. Although the overall combined design is not completely orthogonal, correlations among its input factors would still remain small, since the columns within each of its sub-arrays are still orthogonal or nearly orthogonal to each other. When combining the two stages of arrays, the additional points from the second array can be viewed as

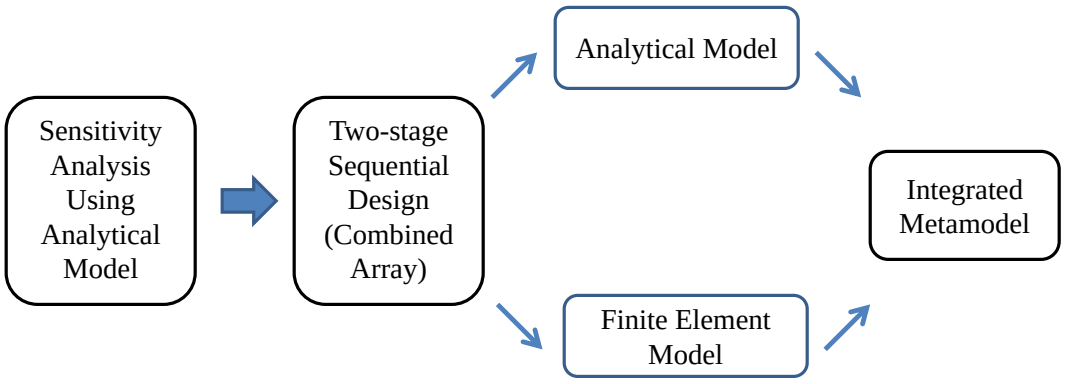

Figure 21: Proposed approach for integrating the analytical models with the finite element models.

filling in the vacant space left by the first array, which can be considered as an attempt to improve the space-filling properties of the design. Moreover, the two-stage design allows a moderate projection redundancy of design points, which makes it more capable in estimating the factor interactions. Since the finite element models in our micromachining process are subject to numerical errors, reducing the number of levels in this way might also help improve the efficiency of estimating the effects. After we obtain data from both the analytical and finite element models for the combined design, many model adjusting techniques can be applied for developing the final surrogate model. A flow chart illustrating our proposed strategy is shown in Figure 21 and a detailed analysis for our micromachining process using the proposed method is offered in next section.

3.4 Analysis

In the micromachining process, we are interested in predicting the machining forces with the input variables of cutting speed V (from 10 to 1000 mm/min), the depth of cut h (from 5 to 100 microns) and the rake angle α_{avg} (from -10 to 10 degrees). For simplicity, we will denote them as x_1, x_2, x_3, respectively. In this section we develop the machining force metamodels step by step according to the proposed flow chart in Figure 21.

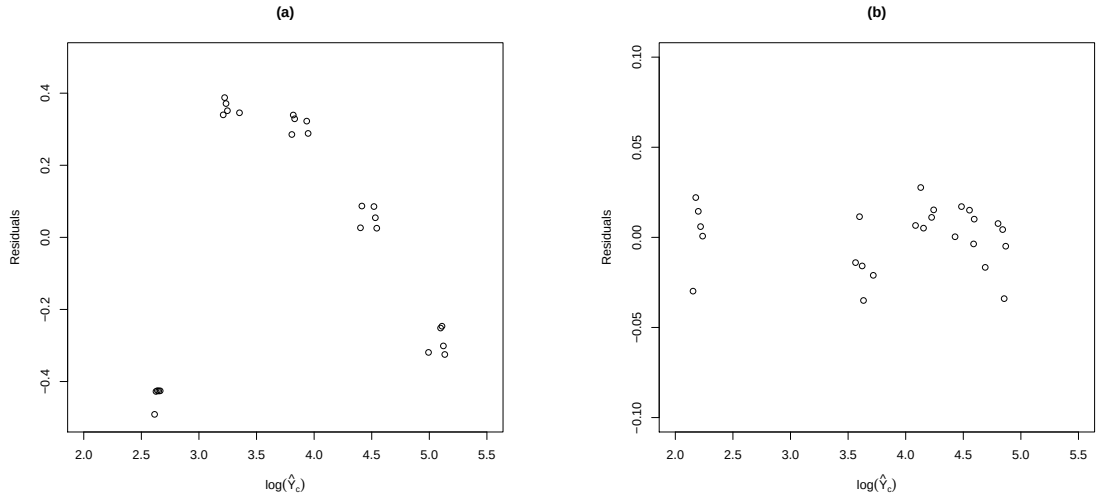

Figure 22: Residuals versus fitted values plots for the cutting force regression model: (a) initial model containing only three linear effects x_1, x_2, x_3; (b) the final fitted model with seven significant terms as shown in Table 3.

3.4.1 Sensitivity Analysis Using Analytical Models

To perform the sensitivity analysis, the analytical models described in the first section are evaluated with a 25-run OA to assess the effects of each input. The design matrix for this 25-run OA is shown in Table 2, which is a 5^{-1} faction of the 125-run five-level full factorial design. In this design, each input factor is assigned with five equally spaced levels to capture the possible high-order effects, and since its three input columns are orthogonal to each other, this OA enables us to independently assess the effects of different input variables in the regression model (Wu and Hamada 2009). For each combination of the input settings, the analytical models return two types of machining forces: the cutting force Y_c and the thrust force Y_t. Outputs of these two machining forces are summarized in Table 2. It is interesting to note that since the analytical models are fast to run, it is also possible to consider using other OAs with larger run size or adopting more sophisticated sensitivity analysis techniques in this step.

Using a residual plot-based regression model building approach, we obtain a model (with R^2 larger than 0.99) for the log values of the cutting force $log(Y_c)$, which contains seven significant terms as shown in Table 3. Figure 22 demonstrates its residual

Table 2: 25-run OA for the sensitivity analysis.

Run	Velocity (x_1)	Depth of cut (x_2)	Rake angle (x_3)	Cutting force (Y_c)	Thrust force (Y_t)
1	10	5	-10	8.3632	5.5678
2	10	28.8	-5	34.8359	19.7924
3	10	52.6	0	59.8586	32.9236
4	10	76.4	5	83.8413	45.2016
5	10	100	10	107.1098	56.9864
6	257.5	5	-5	9.0214	5.9898
7	257.5	28.8	0	36.9887	20.6460
8	257.5	52.6	5	63.9554	34.6482
9	257.5	76.4	10	90.1481	48.0571
10	257.5	100	-10	127.2576	72.1800
11	505	5	0	9.1503	6.0699
12	505	28.8	5	36.8284	20.2381
13	505	52.6	10	64.0589	34.2902
14	505	76.4	-10	99.9650	56.8495
15	505	100	-5	129.4703	73.4350
16	752.5	5	5	9.2548	6.1448
17	752.5	28.8	10	36.5474	19.8182
18	752.5	52.6	-10	70.6782	40.3996
19	752.5	76.4	-5	98.0932	54.6422
20	752.5	100	0	124.0822	67.7131
21	1000	5	10	9.3717	6.2392
22	1000	28.8	-10	40.3503	23.3824
23	1000	52.6	-5	69.1417	38.7190
24	1000	76.4	0	96.4487	52.7727
25	1000	100	5	122.6468	65.9637

diagnostic plots (fitted v.s. residual) before and after incorporating the interaction and nonlinear terms, which shows that finally all the residuals from the fitted model can be contained in a narrow horizontal band around zero. The significant effects identified in fitting the log values of thrust force Y_t are very similar and omitted here. According to the preliminary results in Table 3, the analytical models suggest that x_1 has a quadratic effect, x_2 has a highly nonlinear effect and x_3 has only a linear effect in the machining forces.

Table 3: Significant terms in the regression model for $log(Y_c)$.

	Intercept	$\log(x_2)$	x_1	x_2	x_1:x_2	$x_1^2x_2$	x_3
Estimate	8.56e-01	7.75e-01	1.56e-04	2.95e-03	2.68e-06	-3.25e-09	-3.40e-03
P-values	<2e-16	<2e-16	1.66e-06	3.08e-07	2.82e-03	8.51e-05	3.59e-05

3.4.2 Two-stage Design for the Finite Element Simulations

Based on the results from sensitivity analysis, assigning three levels for x_1, four levels for x_2, and two levels for x_3 seems to be a reasonable strategy to capture the quadratic, nonlinear and linear effects in x_1, x_2 and x_3 respectively. As a result, the Stage-I experiments can be chosen as an OA containing full combinations of these basic levels, which requires conducting $3 \times 4 \times 2 = 24$ runs in computer experiments. Without loss of generality, suppose the range of each factor is coded from 0 to 1. Then the Stage-I experiments select three levels (0, 1/2, 1) for x_1, four levels (0, 1/3, 2/3, 1) for x_2, and two levels (0, 1) for x_3. Because the input effects identified in the sensitivity analysis may have been underestimated, additional levels need to be filled in at the midpoints between the existing basic levels. Specifically, the Stage-II design adds two new levels (1/4, 3/4) for x_1, three new levels (1/6, 3/6, 5/6) for x_2 and one new level (1/2) for x_3. As illustrated in Figure 23, the resulting combined design achieves five levels in x_1, seven levels in x_2 and three levels in x_3, which are large enough to capture the complex nonlinear factor effects in the finite element simulations and are also consistent with the preliminary sensitivity analysis results based on the analytical models. The full combination of all these levels requires $5 \times 7 \times 3 = 105$ simulation runs, which are obviously too much for running the time-consuming finite element simulations. For run-size economy in the computer experiments, a Stage-II design containing full combinations of the added midpoint levels can be constructed, which has $2 \times 3 \times 1 = 6$ runs. After combining the two stages of arrays, the combined design is shown in Table 4, whose total run size is only $24 + 6 = 30$.

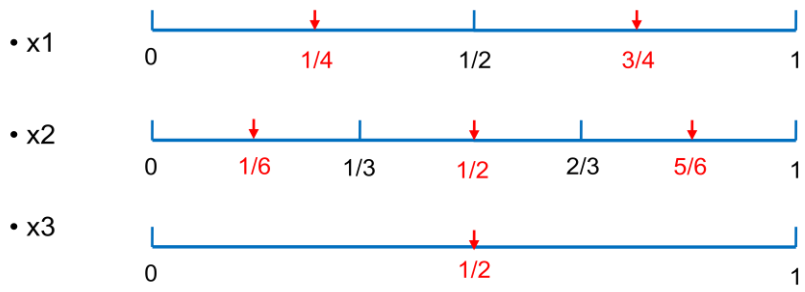

Figure 23: Levels for each input factor.

Table 4: Illustration of the two-stage design.

Stage	Run	x_1	x_2	x_3
I	1	0	0	0
	2	0	0	1
	3	0	0.33	0
	4	0	0.33	1
	5	0	0.67	0
	6	0	0.67	1
	7	0	1	0
	8	0	1	1
	9	0.5	0	0
	10	0.5	0	1
	11	0.5	0.33	0
	12	0.5	0.33	1
	13	0.5	0.67	0
	14	0.5	0.67	1
	15	0.5	1	0
	16	0.5	1	1
	17	1	0	0
	18	1	0	1
	19	1	0.33	0
	20	1	0.33	1
	21	1	0.67	0
	22	1	0.67	1
	23	1	1	0
	24	1	1	1

Stage	Run	x_1	x_2	x_3
II	1	0.25	0.17	0.5
	2	0.25	0.5	0.5
	3	0.25	0.83	0.5
	4	0.75	0.17	0.5
	5	0.75	0.5	0.5
	6	0.75	0.83	0.5

3.4.3 Integrated Metamodel

Based on the 30-run combined design, we ran finite element simulations for the machining forces and their results are shown in Table 5. On a 4GB RAM Intel Core

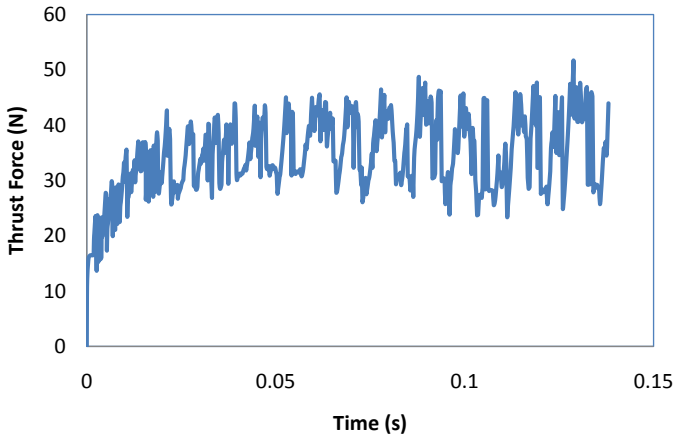

Figure 24: Illustration of numerical fluctuations in the DEFORM® outputs.

2 Duo processor machine, each of these simulation runs can take from 30 minutes to several hours to complete depending on the depth of cut. Since output from the DEFORM® software often involves numerical fluctuations as shown in Figure 24, we report both the mean and standard deviation of each output after the simulation gets stabilized. In the 30th experiment, the cutting force simulation failed to converge properly and its result was omitted from the study. In Table 5, corresponding machining force outputs from the analytical models are also provided.

To synthesize the simulation data with different levels of accuracy, many sophisticated modeling approaches are available in the literature. Kennedy and O'Hagan (2000) proposed an autoregressive structure to link the low-accuracy data (Y_{low}) to high-accuracy data (Y_{high}) through $Y_{high}(\boldsymbol{x}) = \rho Y_{low}(\boldsymbol{x}) + \delta(\boldsymbol{x})$, where a constant ρ is chosen for the scale adjustment and a Gaussian process model $\delta(\boldsymbol{x})$ for location adjustment. Qian et al. (2006) relaxed the constant ρ assumption with a linear regression model for $\rho(\boldsymbol{x})$. Qian and Wu (2008) further developed this model by simultaneously performing the scale and location adjustments using the Gaussian process models. Xia, Ding and Mallick (2011) extended the method to integrate misaligned two-resolution data, and recently Tuo, Wu and Yu (2012) proposed an alternative approach which uses nonstationary Gaussian process models for integrating multi-resolution data.

Among these many choices, in this chapter we adopt an integration framework as

Table 5: Computer outputs of analytical model (AM), finite element simulation (FES) and the standard deviation of simulation error (SD).

Run	x_1	x_2	x_3	Cutting force (Y_c)			Thrust force (Y_t)		
				AM	FES	SD	AM	FES	SD
1	10	5	-10	8.31	9.31	0.650	5.52	6.12	1.625
2	10	5	10	8.28	8.81	1.020	5.49	5.8	1.800
3	10	36.66	-10	44.9	53.8	1.075	25.85	31.7	1.550
4	10	36.66	10	41.01	48.2	1.075	22.09	20.5	1.325
5	10	68.32	-10	80.96	87.4	2.265	46.1	38.6	1.625
6	10	68.32	10	74.24	81.2	2.750	39.61	35.4	1.975
7	10	100	-10	116.58	123.1	2.025	66.12	71.2	1.000
8	10	100	10	107.1	116.6	1.325	56.98	57.4	2.893
9	505	5	-10	9.22	11.2	0.800	6.13	6.16	2.575
10	505	5	10	9.22	10.5	1.000	6.13	5.9	2.650
11	505	36.66	-10	49.73	54.3	1.825	28.63	31.2	1.200
12	505	36.66	10	45.41	47.4	1.388	24.46	26.4	2.475
13	505	68.32	-10	89.81	99.24	1.750	51.14	60.2	1.225
14	505	68.32	10	82.34	92.4	1.575	43.94	41	5.825
15	505	100	-10	129.47	145.6	1.800	73.43	75.9	1.175
16	505	100	10	118.92	127.3	1.525	63.27	69.7	3.325
17	1000	5	-10	9.37	11.8	0.950	6.23	6.22	2.375
18	1000	5	10	9.37	10.9	0.775	6.23	6.12	2.450
19	1000	36.66	-10	50.57	54.9	0.708	29.11	29.8	1.135
20	1000	36.66	10	46.18	51.9	0.800	24.88	22.8	1.823
21	1000	68.32	-10	91.35	102.3	2.000	52.02	57.9	1.675
22	1000	68.32	10	83.75	89.5	2.275	44.69	40.1	8.575
23	1000	100	-10	131.71	142.2	2.750	74.7	69.8	2.950
24	1000	100	10	120.97	132.4	3.525	64.36	59.4	4.400
25	257.5	20.83	0	27.46	31.5	0.875	15.53	14.13	2.085
26	257.5	52.49	0	65.1	75.3	1.275	35.81	42.3	1.525
27	257.5	84.15	0	102.27	109	2.025	55.9	51	1.125
28	752.5	20.83	0	28.2	30.6	0.975	15.95	14.68	1.785
29	752.5	52.49	0	66.87	73.8	2.250	36.78	33.45	2.475
30	752.5	84.15	0	105.08	NA	NA	57.43	51.75	1.418

follows

$$z(\boldsymbol{x}) = \log Y(\boldsymbol{x}) - \log Y_{AM}(\boldsymbol{x}) = \rho(\boldsymbol{x}) + \varepsilon(\boldsymbol{x}), \tag{37}$$

where $Y_{AM}(\boldsymbol{x})$ represents the analytical model, $Y(\boldsymbol{x})$ corresponds to the finite element simulation outputs, $\varepsilon(\boldsymbol{x}) \sim N(0, \sigma_\varepsilon^2(z(\boldsymbol{x}))$ are the random errors uncorrelated at different input locations, and $\rho(\boldsymbol{x})$ is assumed to be a realization from a stationary *Gaussian process* (GP) $\rho(\boldsymbol{x}) \sim GP(\mu, \sigma^2 R)$ which has mean μ and covariance function $\text{cov}(Y(\boldsymbol{x}+\boldsymbol{h}), Y(\boldsymbol{x})) = \sigma^2 R(\boldsymbol{h})$. Here $R(\boldsymbol{h})$ is usually chosen to be the Gaussian correlation function $R(\boldsymbol{h}) = \exp(-\sum_{j=1}^{p} \theta_j h_j^2)$ with unknown correlation parameters $(\theta_1, \ldots, \theta_p)$. Since the machining forces are nonnegative, log values of the corresponding responses are used in the model. The error variances for $z(\boldsymbol{x})$ at each design point can be approximated by $\sigma_\varepsilon^2(z(\boldsymbol{x}_i)) = \sigma_\varepsilon^2(\log Y(\boldsymbol{x}_i)) \approx \sigma_\varepsilon^2(Y(\boldsymbol{x}_i))/E^2(Y(\boldsymbol{x}_i))$, since the analytical model $Y_{AM}(\boldsymbol{x})$ is deterministic. Note that different from Kennedy and O'Hagan (2000) and Qian et al. (2006), we do not use another GP to approximate the low accuracy model because in our case the analytical model is very cheap to evaluate.

In computer experiments' literature, the GP modeling strategy (also referred to as *kriging*) is well studied (Sacks et al. 1989, Santner et al. 2003). Recently, Ankenman et al. (2010) gave a detailed study on the modeling of simulation data with stochastic errors. For a n-run design $D = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$, denote the response values as $\boldsymbol{z} = (z_1, \ldots, z_n)^\top$ and the error variances $\boldsymbol{\Sigma}_\varepsilon = \text{Diag}\{\sigma_\varepsilon^2(z(\boldsymbol{x}_1)), \ldots, \sigma_\varepsilon^2(z(\boldsymbol{x}_n))\}$. The corresponding best linear unbiased predictor (BLUP) at a new input location $\boldsymbol{x}$ can be derived as

$$\hat{z}(\boldsymbol{x}) = \hat{\mu} + \boldsymbol{r}^\top(\boldsymbol{x})(\boldsymbol{R} + \frac{1}{\sigma^2}\boldsymbol{\Sigma}_\varepsilon)^{-1}(\boldsymbol{z} - \hat{\mu}\mathbf{1}), \tag{38}$$

where $\boldsymbol{r}(\boldsymbol{x}) = (R(\boldsymbol{x} - \boldsymbol{x}_1), \ldots, R(\boldsymbol{x} - \boldsymbol{x}_n))^\top$, $\boldsymbol{R}$ is an $n \times n$ matrix with the (ij)th element $R(\boldsymbol{x}_i - \boldsymbol{x}_j)$, $\mathbf{1}$ is a n-dimensional vector of 1's, and $\hat{\mu} = (\mathbf{1}^\top(\boldsymbol{R} + \frac{1}{\sigma^2}\boldsymbol{\Sigma}_\varepsilon)^{-1}\mathbf{1})^{-1}(\mathbf{1}^\top(\boldsymbol{R} + \frac{1}{\sigma^2}\boldsymbol{\Sigma}_\varepsilon)^{-1}\boldsymbol{z})$. The unknown parameters $(\theta_1, \ldots, \theta_p)$ and σ^2 in the predictor can be estimated by maximizing the profile likelihood function, which is equivalent to minimizing

$$\log|\sigma^2\boldsymbol{R} + \boldsymbol{\Sigma}_\varepsilon| + (\boldsymbol{z} - \hat{\mu}\mathbf{1})^\top(\sigma^2\boldsymbol{R} + \boldsymbol{\Sigma}_\varepsilon)^{-1}(\boldsymbol{z} - \hat{\mu}\mathbf{1}). \tag{39}$$

Table 6: Validation data from the analytical model (AM) and the finite element simulations (FES).

Run	x_1	x_2	x_3	Cutting force (Y_c)		Thrust force (Y_t)	
				AM	FES	AM	FES
1	65.00	63.06	2.22	74.15	86.39	40.38	37.98
2	175.00	20.83	-4.44	27.81	25.77	15.97	12.64
3	285.00	84.17	-6.67	105.99	127.21	59.37	57.94
4	395.00	31.39	6.67	39.43	41.86	21.51	22.67
5	505	94.72	4.44	114.63	120.62	61.77	54.53
6	614.99	41.94	-8.89	56.36	65.35	32.19	42.75
7	724.99	10.28	0	15.31	17.21	9.28	9.05
8	834.99	52.50	8.89	64.94	64.22	34.85	29.34
9	944.99	68.61	-2.22	87.75	87.12	48.44	39.16

For our micromachining process, outputs from both the analytical model and finite element simulations are summarized in Table 5, which can be used to fit the GP adjustment model in (37). To facilitate the estimation of correlation parameters, all three input variables are standardized into the range of [0,1] before analysis. By optimizing the likelihood function in (39), we can obtain the estimates for the unknown parameters in the cutting force model as $\hat{\boldsymbol{\theta}} = (8.4, 10, 0.4)$, $\hat{\sigma}^2 = 0.001$; and similarly the parameters for the thrust force model can be estimated to be $\hat{\boldsymbol{\theta}} = (10, 10, 10)$, $\hat{\sigma}^2 = 0.012$. The final surrogate model for the machining forces (at a new input location $\boldsymbol{x}$) is given by

$$\hat{Y}(\boldsymbol{x}) = Y_{AM}(\boldsymbol{x})\exp(\hat{z}(\boldsymbol{x})), \tag{40}$$

where $\hat{z}(\boldsymbol{x})$ is the BLUP in (38) with the above estimated parameters plugged in. It is interesting to note that, the force prediction from this final surrogate model is always equal to zero whenever the output of the analytical model is zero, which agrees with the physical laws.

3.4.4 Validation

To evaluate the effectiveness of our method, we conducted nine additional finite element simulations and compared their results with the predictions from the fitted

surrogate model. These additional experiments are chosen according to an orthogonal-maximin LHD (Joseph and Hung 2008) where each factor has nine equally spaced levels in the input region. The design and the corresponding outputs from both the analytical models and the finite element simulations are shown in Table 6. For each input setting, two different metamodels are fitted for comparison. The first model $\hat{y}_{simp}(\boldsymbol{x})$ is a simple GP model which is fitted only based on the finite element simulation data. The second metamodel $\hat{y}_{int}(\boldsymbol{x})$ is the proposed integrated model in (40) which absorbs the analytical model as the basis and adjusts the prediction by the finite element simulation outputs. The accuracy of these models can be measured and compared by computing their root mean square prediction error (RMSPE) $\sqrt{\sum_{i=1}^{9}[\hat{y}(\boldsymbol{x}_i) - y(\boldsymbol{x}_i)]^2/9}$ on the validation data. For the cutting force prediction, the RMSPE for $\hat{y}_{simp}(\boldsymbol{x})$ is 8.11 while for $\hat{y}_{int}(\boldsymbol{x})$ is 6.57, both of which are much smaller than only using the analytical model for prediction which yields a RMSPE of 9.01. Specifically, the proposed integrated model $\hat{y}_{int}(\boldsymbol{x})$ is 27.1% more accurate than the analytical model, and it also improves the prediction accuracy by 19% over the traditional $\hat{y}_{simp}(\boldsymbol{x})$ model. Note that in Figure 5, since the analytical models always give a cutting force smaller than the finite element models, their ratios can be less variable than the finite element simulation outputs, which partly explains the reason for the superior performance of our integrated model $\hat{y}_{int}(\boldsymbol{x})$ here. For predicting the thrust force, the RMSPE of only using the approximate analytical model is 5.78. When a simple GP model is fitted based on the detailed finite element simulation data, it gives a RMSPE of 5.50, which does not make too much improvement. Clearly, although the fitted $\hat{y}_{simp}(\boldsymbol{x})$ is precise at the design points, its prediction accuracy decreases when moving into the unknown region. By integrating the analytical model with the finite element simulation data, our final metamodel $\hat{y}_{int}(\boldsymbol{x})$ for the thrust force can give a RMSPE as low as 4.72, which is 18.3% and 14.2% more accurate than using the analytical model or the GP model based on the finite element simulation data alone.

3.5 Conclusion

In this chapter, we discuss strategies to efficiently integrate information from analytical models with finite element models in building metamodels in a micromachining process. We show that the cheap analytical models can be used to perform a sensitivity analysis in the beginning, whose results can then guide us to more efficiently design the computationally expensive finite element simulations. A two-stage design for the finite element simulations is proposed which makes use of the elicited prior knowledge from the analytical models and assigns a customized number of levels for each input factor. The proposed design is comprised of two sub-arrays and enjoys several advantages such as near-orthogonality, good space-filling properties and desirable ability in estimating both the nonlinear effects and factor interactions. For simulation with numerical errors, the proposed design is also more efficient of estimating the effects compared to the traditional n-level space-filling designs. In the end, our model validation results indicate that the integrated model can more accurately approximate the overall response surface than either using the analytical model or the traditional metamodel based on the finite element simulations alone. Since the integrated metamodel is much faster to evaluate than the finite element simulations, it can be used to better understand, optimize, and control the micromachining process.

In this case study, we are a little fortunate that the sensitivity analysis for both the cutting forces and the thrust forces yield similar results. Since these two forces are two simultaneous outputs from the finite element models, designs for them always have to be identical. In some applications, if the two responses turn out to have very different sensitivity with respect to the inputs, we need to construct their design based on the more complex effects for each of their input factors. In addition, although in this chapter we focus on extracting information from analytical models in developing the integrated metamodel, the approach may be more broadly applicable to other types of engineering models which are faster but less accurate than the finite element simulations. For example, besides the analytical models, it is possible to have other approximate engineering models such as the ones involving only ordinary differential

equations but requiring numerical methods to solve them, or those needing some nonlinear root-finding methods, or those using finite difference methods, etc. These approximate models can be used to design the expensive finite element simulations in a similar way and finally build a fast and accurate integrated surrogate model.

REFERENCES

Ababou, R., Bagtzoglou, A. C., and Wood, E. F. (1994), "On the Condition Number of Covariance Matrices in Kriging, Estimation, and Simulation of Random Fields," *Mathematical Geology*, 26, 99-133.

Ankenman, B., Nelson, B. L., and Staum, J. (2010), "Stochastic Kriging for Simulation Metamodeling," *Operations Research*, 58, 371-382.

Anderes, E. B., and Stein, M. L. (2008), "Estimating Deformations of Isotropic Gaussian Random Fields on the Plane," *Annals of Statistics*, 36, 719-741.

Ba, S., and Joseph, V. R. (2011), "Multi-Layer Designs for Computer Experiments," *Journal of the American Statistical Association*, 106, 1139-1149.

Banerjee, S., Charlin, B. P., and Gelfand, A. E. (2003), *Hierarchical Modeling and Analysis for Spatial Data*, Florida: Chapman and Hall/CRC.

Bingham, D., Sitter, R. R., and Tang, B. (2009), "Orthogonal and Nearly Orthogonal Designs for Computer Experiments," *Biometrika*, 96, 51-65.

Box, G. E. P., and Hunter, J. S. (1961), "The 2^{k-p} Fractional Factorial Designs," *Technometrics*, 3, 311-351 and 449-458.

Box, G. E. P., Hunter, J. S., and Hunter, W. G. (2005), *Statistics for Experiments*: *Design, Innovation, and Discovery* (2nd ed.), New York: Wiley.

Challen, J. M., and Oxley, P. L. B. (1984), "Slip-Line Fields for Explaining the Mechanics of Polishing and Related Processes," *Int. J. Mech. Sci.*, 26, 403-418.

Cressie, N. A. (1991), *Statistics for Spatial Data*, New York: Wiley.

Currin, C., Mitchell, T. J., Morris, M. D. and Ylvisaker, D. (1991), "Bayesian Prediction of Deterministic Functions, with Applications to the Design and Analysis of Computer Experiments," *Journal of the American Statistical Association*, 86, 953-963.

Fang, K. T., Li, R., and Sudjianto, A. (2006), *Design and Modeling for Computer Experiments*, London: Chapman and Hall.

Fang, K. T., and Mukerjee, R. (2000), "A Connection Between Uniformity and Aberration in Regular Fractions of Two-Level Factorials," *Biometrika*, 87, 193-198.

Fries, A., and Hunter, W. G. (1980), "Minimum Aberration 2^{k-p} Designs," *Technometrics*, 22, 601-608.

Gneiting, T., and Raftery, A. E. (2007), "Strictly Proper Scoring Rules, Prediction, and Estimation," *Journal of the American Statistical Association*, 102, 359-378.

Gramacy, R. B., and Lee, H. K. H. (2008), "Bayesian Treed Gaussian Process Models with an Application to Computer Modeling," *Journal of the American Statistical Association*, 103, 1119-1130.

Gramacy, R. B., and Lee, H. K. H. (2011), "Cases for the Nugget in Modeling Computer Experiments," *Statistics and Computing*, to appear.

Haaland, B., and Qian, P. Z. G. (2011), "Accurate Emulators for Large-Scale Computer Experiments," *Annals of Statistics*, 39, 2974-3002.

Hedayat, A., Sloane, N., and Stufken, J. (1999), *Orthogonal Arrays*, New York: Springer Verlag.

Higdon, D. M., Swall, J., and Kern, J. (1999), "Non-Stationary Spatial Modeling," *Bayesian Statistics 6, Proceedings of the Sixth Valencia International Meeting*, 761-768, Oxford University Press.

Huang, W., Wang, K., Breidt, F. J., and Davis, R. A. (2011), "A Class of Stochastic Volatility Models for Environmental Applications," *Journal of Time Series Analysis*, 32, 364-377.

Jin, R., Chen, W., and Sudjianto, A. (2005), "An Efficient Algorithm for Constructing Optimal Design of Computer Experiments," *Journal of Statistical Planning and Inference*, 134, 268-287.

Johnson, M. E., Moore, L. M., and Ylvisaker, D. (1990), "Minimax and Maximin Distance Designs," *Journal of Statistical Planning and Inference*, 26, 131-148.

Joseph, V. R. (2006), "Limit Kriging," *Technometrics*, 48, 458-466.

Joseph, V. R., and Hung, Y. (2008), "Orthogonal-Maximin Latin Hypercube Designs," *Statistica Sinica*, 18, 171-186.

Joseph, V. R., Hung, Y., and Sudjianto, A. (2008), "Blind Kriging: A New Method for Developing Metamodels," *ASME Journal of Mechanical Design*, 130, 031102 (8 pages).

Joseph, V. R., and Kang, L. (2011), "Regression-Based Inverse Distance Weighting with Applications to Computer Experiments," *Technometrics*, 53, 254-265.

Kennedy, M. C., and O'Hagan, A. (2000), "Predicting the Output From a Complex Computer Code When Fast Approximation Are Available," *Biometrika*, 87, 1-13.

Kerr, M. K. (2001), "Bayesian Optimal Fractional Factorials," *Statistica Sinica*, 11, 605-630.

Kiefer, J. C. (1975), "Construction and Optimality of Generalized Youden Designs," *A Survey of Statistical Design and Linear Models* (J. N. Srivastava, ed.), 333-353, Amsterdam: North-Holland.

Kwong, K. Y. (2004), "Two-Level Maximin Distance Fractional Factorial Designs," *UCLA Statistics Theses* [online: http://theses.stat.ucla.edu/25/thesis.pdf].

Li, W., and Lin, D. K. J. (2003), "Optimal Foldover Plans for Two-Level Fractional Factorial Designs," *Technometrics*, 45, 142-149.

Lin, C. D., Bingham, D., Sitter, R. R., and Tang, B. (2010), "A New and Flexible Method for Constructing Designs for Computer Experiments," *Annals of Statistics*, 38, 1460-1477.

Manjunathaiah, J., and Endres, W. J. (2000), "A New Model and Analysis of Orthogonal Machining With an Edge-Radiused Tool," *ASME J. Manuf. Sci. Eng.*, 122, 384-390.

McKay, M. D., Beckman, R. J., and Conover, W. J. (1979), "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output From a Computer Code," *Technometrics*, 21, 239-245.

Merchant, M. E. (1944), "Basic Mechanics of the Metal-Cutting Process," *Journal of Applied Mechanics*, 11, A168-A175.

Morris, M. D., and Mitchell, T. J. (1995), "Exploratory Designs for Computational Experiments," *Journal of Statistical Planning and Inference*, 43, 381-402.

Morris, M. D., Mitchell, T. J., and Ylvisaker, D. (1993), "Bayesian Design and Analysis of Computer Experiments: Use of Derivatives in Surface Prediction," *Technometrics*, 35, 243-255.

Mukerjee, R., and Wu, C. F. J. (2006), *A Modern Theory of Factorial Design*, New York: Springer.

Owen, A. B. (1992), "Orthogonal Arrays for Computer Experiments, Integration and Visualization," *Statistica Sinica*, 2, 439-452.

Paciorek, C., and Schervish, M. (2006), "Spatial Modelling Using a New Class of Nonstationary Covariance Functions," *Environmetrics*, 17, 483-506.

Peng, C. Y., and Wu, C. F. J. (2012), "Regularized Kriging," submitted.

Piepel, G. F., Anderson, C. M., and Redgate, P. E. (1993), "Response Surface Designs for Irregularly-Shaped Regions (Parts 1, 2 and 3)," *1993 Proceedings of the Section on Physical and Engineering Sciences*, 205-227, American Statistical Association, Alexandria, VA.

Qian, P. Z. G. (2009), "Nested Latin Hypercube Designs," *Biometrika*, 96, 957-970.

Qian, P. Z. G., Ai, M., and Wu, C. F. J. (2009a), "Construction of Nested Space-Filling Designs," *Annals of Statistics*, 37, 3616-43.

Qian, P. Z. G., Seepersad, C. C., Joseph, V. R., Allen, J. K., and Wu, C. F. J. (2006), "Building Surrogate Models with Detailed and Approximate Simulations," *ASME Journal of Mechanical Design*, 128, 668-677.

Qian, P. Z. G., Tang, B., and Wu, C. F. J. (2009b), "Nested Space-Filling Designs for Experiments With Two Levels of Accuracy," *Statistica Sinica*, 19, 287-300.

Qian, P. Z. G., and Wu, C. F. J. (2008), "Bayesian Hierarchical Modeling for Integrating Low-Accuracy and High-Accuracy Experiments," *Technometrics*, 50, 192-204.

Ranjan, P., Haynes, R., and Karsten, R. (2012), "Gaussian Process Models and Interpolators for Deterministic Computer Simulators," *Technometrics*, to appear.

Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P. (1989), "Design and Analysis of Computer Experiments," *Statistical Science*, 4, 409-423.

Sampson, P. D., and Guttorp, P. (1992), "Nonparametric Estimation of Nonstationary Spatial Covariance Structure," *Journal of the American Statistical Association*, 87, 108-119.

Santner, T. J., Williams, B. J., and Notz, W. (2003), *The Design and Analysis of Computer Experiments*, New York: Springer.

Saltelli, A., Chan, K., and Scott, E. (2000), *Sensitivity Analysis*, John Wiley & Sons, Chichester.

SAS Institute Inc. (2008), *JMP User Guide* (Release 8), Cary, NC: Author.

Schmidt, A. M., and O'Hagan, A. (2003), "Bayesian Inference for Nonstationary Spatial Covariance Structure via Spatial Deformations," *Journal of the Royal Statistical Society, Series B*, 65, 745-758.

Shepard, D. (1968), "A Two-Dimensional Interpolation Function for Irregularly-Spaced Data," *Proceedings of the 1968 ACM National Conference*, 517-524.

Singh, R., and Melkote, S. N. (2009), "Force Modeling in Laser Assisted Micromachining Including the Effect of Machine Deflection," *ASME J. Manuf. Sci. Eng.*, 131, 011013-011022.

Steinberg, D. M., and Lin, D. K. J. (2006), "A Construction Method for Orthogonal Latin Hypercube Designs," *Biometrika*, 93, 279-288.

Tang, B. (1993), "Orthogonal Array-Based Latin Hypercubes," *Journal of the American Statistical Association*, 88, 1392-1397.

Tuo, R., Wu, C. F. J., and Yu, D. (2011), "Modeling Mesh Density in Computer Experiments," Submitted.

Wackernagel, H. (2003), *Multivariate geostatistics* (3rd ed.), New York: Springer-Verlag.

Waldorf, D. J., DeVor, R. E., and Kapoor, S. G. (1998), "A Slip-Line Field for Ploughing During Orthogonal Cutting," *ASME J. Manuf. Sci. Eng.*, 120, 693-699.

Welch, W. J., Buck, R. J., Sacks, J., Wynn, H. P., Mitchell, T. J., and Morris, M. D. (1992), "Screening, Predicting, and Computer Experiments," *Technometrics*, 34, 15-25.

Wu, C. F. J., and Hamada, M. S. (2009), *Experiments*: *Planning, Analysis, and Optimization* (2nd ed.), New York: Wiley.

Xia, H., Ding, Y., and Mallick, B. (2011), "Bayesian Hierarchical Models for Combining Misaligned Two-Resolution Metrology Data," *IIE Transactions*, 43, 242-258.

Xiong, Y., Chen, W., Apley, D. W., and Ding, X. (2007), "A Non-Stationary Covariance-Based Kriging Method for Metamodelling in Engineering Design," *International Journal for Numerical Methods in Engineering*, 71, 733-756.

Yamamoto, J. K. (2000), "An Alternative Measure of the Reliability of Ordinary Kriging Estimates," *Mathematical Geology*, 32, 430-439.

Yan, H., Hua, J., and Shivpuri, R. (2007), "Flow Stress of AISI H13 Die Steel in Hard Machining," *Materials and Design*, 28, 272-277.

Ye, K. Q. (1998), "Orthogonal Column Latin Hypercubes and Their Application in Computer Experiments," *Journal of the American Statistical Association*, 93, 1430-1439.